DISPOSITIVOS DE MONITORIZACIÓN DE SEÑALES BIOMÉDICAS EMPLEANDO SISTEMAS EMPOTRADOS DE BAJO COSTE

Tiago Frederico Ferreira Gonçalves

Trabajo de fin de grado

Grado en Ingeniería Informática

Facultad de Informática

Universidad Complutense de Madrid

Curso 2018 / 2019

Director

Joaquín Recas Piorno

Agradecimientos

A mi familia, quien me ha animado en este periodo de aprendizaje y gran trabajo,de quien he recibido su apoyo durante toda esta etapa y ha hecho posible que me sienta orgulloso de mi esfuerzo.
A mi director de proyecto, Joaquín, quien me dio la oportunidad de llevar adelante este trabajo y cedió parte del material utilizado.

Resumen

El propósito de este proyecto es crear un sistema de monitorización de señales biomédicas para personas que se exponen a un alto nivel de estrés y esfuerzo en sus funciones profesionales como los pilotos de aviones de combate y pilotos de Formula 1 entre otros. Estos profesionales precisan de una mucha concentración en sus funciones mientras están sometidos a un gran esfuerzo, además, un nivel alto de estrés podría hacerles cometer un error fatal.

Para controlar el estado de estos profesionales, se hará la monitorización de diferentes señales, lo primordial es monitorizar el estado cardiaco, para ello se usa un **electrocardiográma (ECG)**, esta señal es un claro indicador del esfuerzo al que está siendo sometido, además se complementa con una **capnografía** para medir el nivel de **CO_2 en aire exhalado**, con ello además de complementar el **ECG** para saber al esfuerzo que se somete, se puede alertar de alguna insuficiencia respiratoria, en conjunto con estos dos se examina la actividad **electrodérmica (EDA)**, con ello podemos observar cambios en los niveles de sudoración inusuales provocados por el **sistema nervioso autónomo (SNA)** ante una situación de estrés y por último una sonda que mida la **temperatura corporal**, con ello, se podría detectar altas temperaturas que podrían deberse a un golpe de calor y ello provocar un desvanecimiento poniendo en peligro su vida.

Palabras clave: **ECG, CO_2, EDA, Capnografía, Sistema empotrado, Arduino, señal, sensor**

Abstract

The purpose of this project is to create a monitoring system for biomedical signals for people who are exposed to a high level of stress and effort in their professional functions such as fighter pilots and Formula 1 pilots among others. These professionals need a lot of concentration in their functions while they are subjected to a great effort, in addition, a high level of stress could make a fatal error.

To monitor the condition of these professionals a control of different signals will be done, focusing mainly on the cardiac status through the use of an **electrocardiogram** (**ECG**). This signal is a clear indicator of the effort to which the professional is being subjected. It is complemented with a **capnography** to measure the level of **CO_2 in exhaled air**, with this in addition to complement the ECG to know the effort, it may indicate respiratory problems. In conjunction with these, **electrodermal activity** (**EDA**) is examined. thanks to this we can observe changes in the unusual levels of sweating caused by the **autonomic nervous system** (**SNA**) before a stress situation and finally a probe that measures the **body temperature**, this can help us detect high temperatures that could be due to a heatstroke and this cause a fading endangering your life.

Keywords: **ECG, CO_2, EDA, capnography, embedded system, Arduino, signal, sensor**

Índice general

Índice de figuras

Capítulo 1

Introducción

En este trabajo se pretende crear un sistema de monitorización de señales biomédicas para profesionales con grandes niveles de esfuerzo, donde un fallo podría ser fatal.
Para el desarrollo de este sistema se van a realizar las siguientes pruebas: electrocardiograma, niveles de CO_2 en aire exhalado, actividad electrodérmica y temperatura corporal.
Para ello se usarán placas de Arduino y los siguientes sensores: AD8232 4.1.4 para realizar el electrocardiograma, ExplorIR-W 4.1.6 para la capnografía, unos electrodos conectados a la placa para ver la actividad electrodérmica y la sonda ds18b20 4.1.7 para medir la temperatura corporal. Para finalizar se envía la información a una Raspberry y esta lo muestra a través de una interfaz gráfica.

1.1. Motivación

La informática a día de hoy está presente en todos los sectores; si nos paramos a contar la cantidad de elementos que usamos en el día a día y en nuestros trabajos, seguramente nos sorprenderíamos. El mundo ha cambiado totalmente y es posible que cualquier trabajador pudiera mejorar sus labores gracias a esto, algo que para mí es maravilloso, ya que me permite trabajar en diferentes sectores realizando proyectos muy diversos. En este caso, esta idea nace del interés por usar la informática junto a otras disciplinas de la ciencia. Mi especial interés hacia la medicina, hizo que me decantara por un proyecto que relacionase ambas y en consecuencia, me permitiera aprender sobre una nueva rama, dando un salto a los sistemas empotrados y el uso de sensores creando un nuevo interfaz lleno de posibilidades entre el mundo real y los conocimientos obtenidos hasta ahora.
Tenía una idea muy clara acerca de la temática a desarrollar en el proyecto, me parecía interesante, diferente y además me brindaría la oportunidad de ampliar mis conocimientos. Tras trasmitir mi idea a Joaquín, el que más adelante sería el director del proyecto, decidimos utilizar diferentes sensores para obtener señales biomédicas, cumpliendo las expectativas del trabajo que deseaba realizar.

1.2. Objetivos

El objetivo de este proyecto es desarrollar un monitor de diferentes señales biomédicas obtenidas a través de sensores conectados a un Arduino y enviarlas a una Raspberry para mostrar la información. Las señales a monitorizar son las siguientes:

- **ECG:** Recoger en tiempo real la señal cardíaca de una persona o de un simulador de paciente, calcular los BPM y graficar el electrocardiograma.
- **CO_2:** Medir los niveles de CO_2 en aire exhalado en una escala de partes por millón (ppm), mostrar la información obtenida y realizar un gráfico.
- **Temperatura Corporal:** Usando una sonda pegada al cuerpo y conectada al sistema empotrado, recoger la temperatura del paciente y mostrar los datos obtenidos en grados Celsius (^{o}C).
- **EDA:** Obtener la actividad electrodérmica. El objetivo es detectar cambios en la conductividad de la piel que indiquen distintos niveles de sudoración que alerten de una situación inusual.

1.3. Plan de trabajo

Una vez marcados los objetivos se han seguido las siguientes tareas:

1. Búsqueda y lectura de documentación para la selección de los sistemas empotrados a utilizar.
2. Estudio de viabilidad y alcance de los diferentes objetivos marcados.
3. Formación de uso en las diferentes placas de Arduino a emplear.
4. Aprendizaje sobre el electrocardiograma, su funcionamiento, sus usos e interpretación de resultados.
5. Investigación acerca de la placa *AD8232 Heart Monitor*.
6. Utilización del simulador cardiaco *TechPatient Cardio*.
7. Programar placa Arduino conectada con la placa *AD8232*.
8. Redactar breve borrador con el proceso realizado y los resultados obtenidos.
9. Formación sobre capnografía, así como sus usos y lectura de resultados.
10. Estudio acerca del sensor *ExplorIR-W CO2 Sensor* para adaptación al proyecto.
11. Programar placa Arduino conectada con el sensor *ExplorIR-W*.
12. Realizar un resumen con el proceso realizado y sus resultados alcanzados.
13. Obtener de patologías relacionadas con la temperatura corporal.
14. Lectura y estudio de la documentación de la sonda de temperatura *DS18b20*.
15. Programar placa Arduino conectada con la sonda *DS18b20*.
16. Escribir un borrador con el proceso realizado y resultados logrados.
17. Recoger información relacionada con la actividad electrodérmica.
18. Desarrollar un método para obtener la actividad electrodérmica.
19. Anotar los resultados obtenidos y el proceso realizado.
20. Unificar el desarrollo de todos los sensores en una sola placa de Arduino.
21. Crear una interfaz gráfica que muestre la información monitorizada de todas las señales.
22. Unificación de los borradores y desarrollo de la memoria completa.

Capítulo 2

Introduction

This work aims to create a biomedical signal monitoring system for professionals with high levels of estress, where a failure could be fatal. For the development of this system, the following tests will be carried out: electrocardiogram, CO_2 levels in exhaled air, electrodermal activity and body temperature. For this, Arduino boards and the following sensors will be used: AD8232 to perform the electrocardiogram, ExplorIR-W for capnography, electrodes connected to the Arduino board to see the electrodermal activity and the ds18b20 probe to measure body temperature. Finally, the information is sent to a Raspberry and it shows it through a graphical interface.

2.1. Motivation

Today's computer science is there in all sectors; if we count the amount of elements we use on a daily basis and in our work, we would surely be surprised. The world has changed completely and it is possible that any worker could improve their work thank to this, something wonderful to me, since it allows me to work in different sectors carrying out very diverse projects. In this case, this idea is born from the interest to use computer science together with other disciplines of science. My special interest in medicine, made me opt for a project that relates both and consequently, allowed me to learn about a new branch, making a leap to embedded systems and the use of sensors creating a new interface full of possibilities between the real world and the knowledge obtained so far.
I had a very clear idea about the theme to develop in the project, it seemed interesting, different and also it would give me the opportunity to expand my knowledge.
After transmitting my idea to Joaquín, who would later be the project director, we decided to use different sensors to obtain biomedical signals, fulfilling the expectations of the work we wanted to perform

2.2. Objectives

The goal of this project is to develop a monitor of different biomedical signals obtained through sensors connected to an Arduino and send them to a Raspberry in order to show the information. The signals to monitor are the following:

- **ECG:**Collect in real time the cardiac signal of a person or a patient simulator, calculate the BPM and graph the electrocardiogram.
- **CO_2:** Measure the levels of CO_2 in exhaled air on a scale of parts per million (ppm), show the information obtained and make a graph.
- **Body temperature:** Using a probe attached to the body and connected to the embedded system, collect the patient's temperature and display the data obtained in degrees Celsius (ºC).
- **GRS:** Obtain the galvanic response skin. The objective is to detect changes in the conductivity of the skin that indicate different levels of sweating that will alert of an unusual situation.

2.3. Workplan

Once the objectives have been marked, the following tasks have been followed:

1. Researching and reading documentation for the selection of embedded systems to be used.
2. Feasibility study and scope of the different objectives set.
3. Use training in the different Arduino boards to be used.
4. Learning about the electrocardiogram, its operation, its uses and interpretation of results.
5. Research about the sensor *AD8232 Heart Monitor*.
6. Use of the cardiac simulator *TechPatient Cardio*.
7. Program Arduino board connected to the board *AD8232*.
8. Compile of a brief draft with the process carried out and results obtained.
9. Training on capnography, as well as its uses and reading results.
10. Study about the sensor *ExplorIR-W CO2 Sensor* to adapt its functions to the project.
11. Program Arduino board connected with the *ExplorIR-W* sensor.
12. Make a summary of the process performed and its results achieved.
13. Obtain pathologies related to body temperature.
14. Reading and studying of the documentation of the temperature probe *DS18b20*.
15. Program Arduino board connected with the probe *DS18b20*.
16. Write a draft with the process and the results obtained.
17. Collect information related to the electrodermal activity.
18. Develop a method to obtain the electrodermal activity.
19. Record the results obtained and the process performed.
20. Unify the development of all sensors on a single Arduino board.
21. Create a graphical interface that shows the monitored information of all the signals.
22. Unification of the drafts and development of the complete memory.

Capítulo 3

Antecedentes

3.1. Electrocardiograma

Un electrocardiograma (**ECG**) representa gráficamente la actividad eléctrica que se produce en cada latido cardiaco. Esta actividad eléctrica se obtiene de electrodos adheridos a la superficie corporal del paciente y dibuja una representación gráfica o trazado, donde se observan diferentes ondas que representan los estímulos eléctricos de las aurículas y los ventrículos. El uso del **ECG** frente a otras pruebas de medición cardiacas, como un pulsómetro, es la mayor cantidad de información que se puede obtener de las señales eléctricas frente a únicamente el pulso, para detectar enfermedades del corazón.

3.1.1. Derivaciones cardiacas del Electrocardiograma

Las derivaciones cardiacas son el registro de la diferencia de potenciales eléctricos entre dos puntos, ya sea entre dos electrodoshttps://es.overleaf.com/project/5cc89cdaab992b58cf084abe (derivación bipolar) o entre un punto virtual y un electrodo (derivaciones monopolares).
Las derivaciones clásicas o bipolares, registran la diferencia de potencial entre dos electrodos ubicados en extremidades diferentes.

- D1 ó I: mide la diferencia de potencial entre el electrodo del brazo derecho y el izquierdo.
- D2 ó II: mide la diferencia de potencial entre el electrodo del brazo derecho a la pierna derecha.
- D3 ó III: mide la diferencia de potencial entre el electrodo del brazo izquierdo a la pierna derecha.

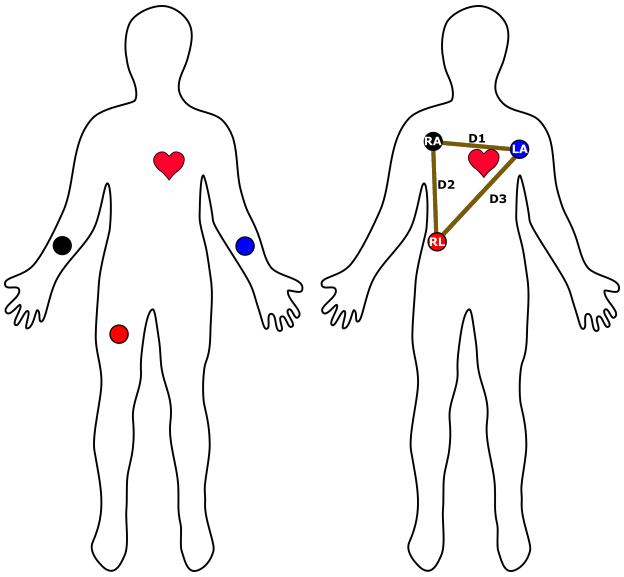


Figura 3.1: *Derivaciones bipolares de extremidades.(Sparkfun[1])*

Las tres derivaciones bipolares forman, en su conjunto, lo que se denomina el *triángulo de Einthoven* ver figura 3.1.

3.1.2. Interpretación de un electrocardiograma

Para comprender a grandes rasgos la lectura de un electrocardiograma, explicaremos los conceptos esenciales y el significado de las ondas generadas.
El ECG se divide en dos intervalos básicos, el **intervalo PR** y el **intervalo QT** (Figura 3.2)
EL **intervalo PR** es la onda inicial generada por un impulso eléctrico que se desplaza desde la aurícula derecha hacia la izquierda. La aurícula derecha recibe un impulso eléctrico hace que las cámaras se despolaricen y obliga a que se contraiga y drene la sangre desoxigenada de la vena cava superior e inferior al ventrículo derecho. Cuando el impulso eléctrico viaja a través de la parte superior del corazón, activa la aurícula izquierda para contraerse. La aurícula izquierda es responsable de recibir sangre recién oxigenada de los pulmones al ventrículo izquierdo a través de las venas pulmonares izquierda y derecha.

En el **intervalo QT** sucede el proceso **QRS**, que es la representación gráfica de la despolarización de los ventrículos del corazón. Durante el **QRS** ambos ventrículos comienzan a bombear. El ventrículo derecho bombea sangre desoxigenada a los pulmones a través de las arterias pulmonares izquierda y derecha, mientras que el ventrículo izquierdo bombea sangre recién oxigenada a través de la aorta y al resto del cuerpo. Después de la contracción inicial viene el segmento **ST** que es bastante silencioso eléctricamente, ya que es el momento hasta que los ventrículos vuelves a ser polarizados. Por último, la onda **T** representa en el momento de relajación de los ventrículos.

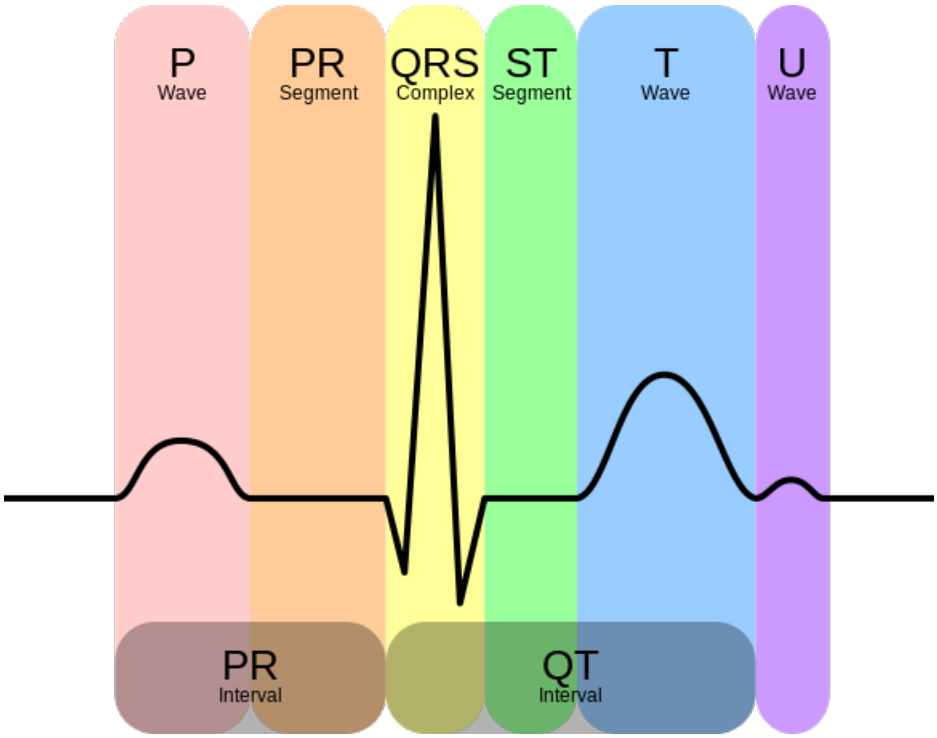


Figura 3.2: *Representación de un ECG etiquetado con de ondas e intervalos (Wikipedia[11])*

Para más información consultar referencias.[11 1]

En la figura 3.2 podemos ver la representación de un electrocardiograma con los tipos de onda y los intervalos descritos en el este apartado.

3.2. Capnografía

3.2.1. ¿Qué es la capnografía?

La **capnografía** es la monitorización no invasiva de concentraciones de dióxido de carbono ($\mathbf{CO_2}$) en la vía aérea de un paciente durante su ciclo respiratorio. Esta aporta un importante papel en la detección temprana de enfermedades respiratorias y en el control del ciclo de eliminación del $\mathbf{CO_2}$ durante intervenciones médicas que requieran del uso de anestesia.

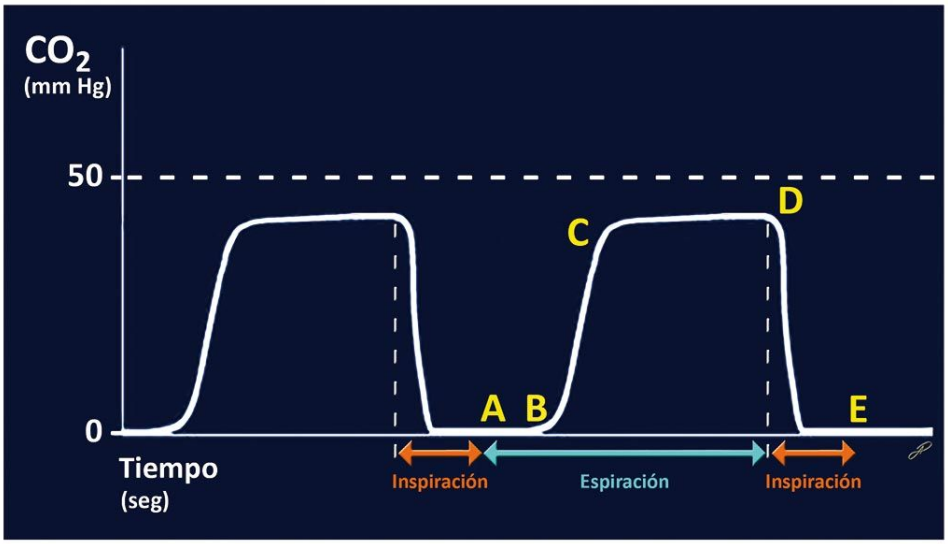


Figura 3.3: *Fases respiratorias en una capnografía.(Zonates.com[13])*

3.2.2. Intercambio de gases en la respiración

Las células de nuestro organismo necesitan un aporte continuo de oxígeno ($\mathbf{O_2}$) para llevar a cabo la respiración celular. El sistema respiratorio es el conjunto de órganos que permite el intercambio de gases, en este procedimiento se elimina el dióxido de carbono ($\mathbf{CO_2}$) generado por las células y las alimenta de oxígeno ($\mathbf{O_2}$) .

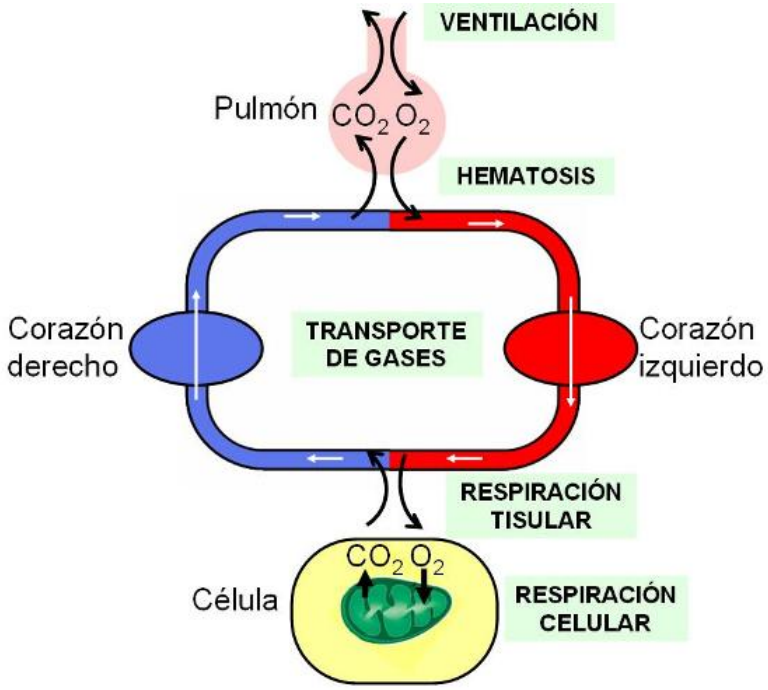


Figura 3.4: *Intercambio de gases en el ciclo respiratorio. (Genomasur.com[3])*

Los procesos que se realizan durante el desarrollo del ciclo respiratorio son los siguientes:

- **Ventilación:** flujo de entrada y salida de aire entre el exterior y los pulmones.
- **Hematosis o respiración externa:** propagación de oxígeno ($\mathbf{O_2}$) y dióxido de carbono ($\mathbf{CO_2}$) entre los alvéolos pulmonares y la sangre.
- **Transporte de gases en sangre:** traslado de oxígeno ($\mathbf{O_2}$) desde los pulmones hasta las células y de dióxido de carbono ($\mathbf{CO_2}$) desde las células hasta los pulmones.
- **Respiración interna o tisular:** propagación de oxígeno ($\mathbf{O_2}$) y de dióxido de carbono ($\mathbf{CO_2}$) entre la sangre y los tejidos.
- **Respiración celular:** proceso por el cual las células degradan las moléculas de alimento por oxidación para obtener energía.

3.2.3. Usos clínicos de la capnografía

Esta monitorización en conjunto con la pulsiometría nos permitirá detectar precozmente los problemas respiratorios graves que surjan durante la asistencia sanitaria como la apnea, obstrucción de la vía respiratoria o hipoventilación del paciente. La detección de estos eventos graves puede demorarse (incluso entre 2-4 minutos) cuando se emplea únicamente pulsioximetría.

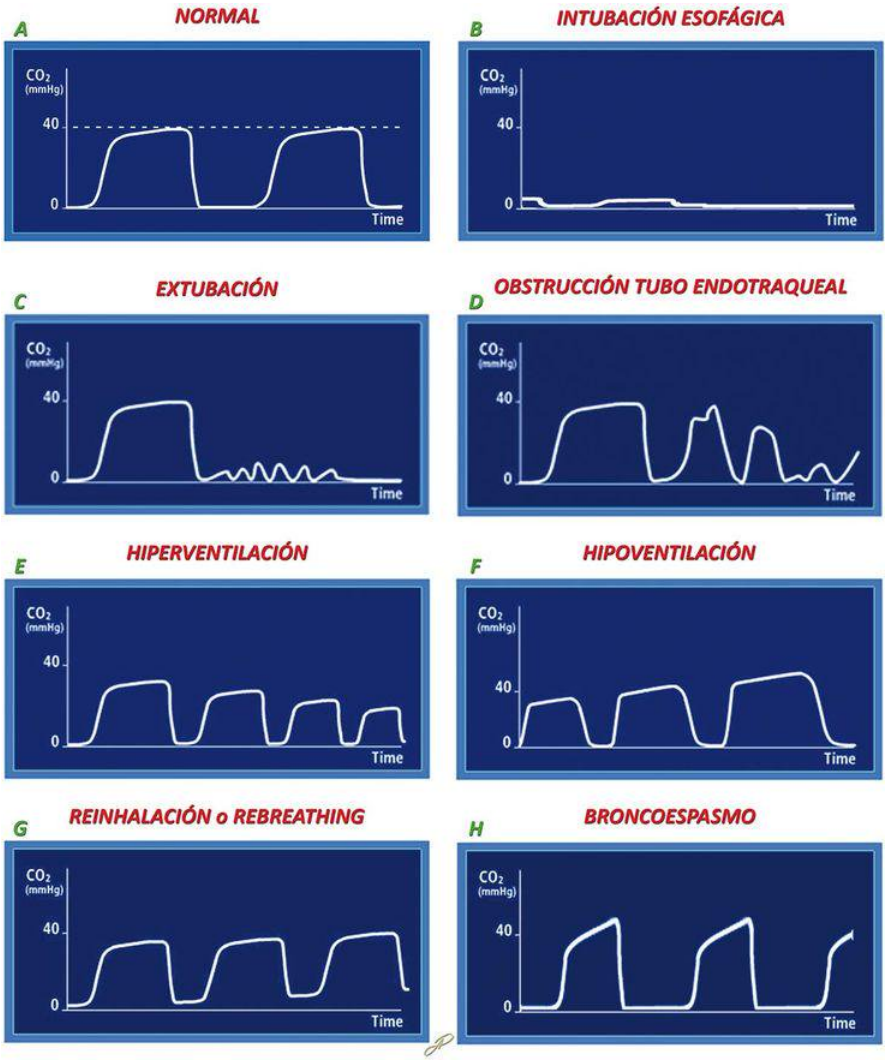


Figura 3.5: *Alteraciones más comunes en capnografías (Zonates.com[13])*

En la figura 3.5 podemos ver lecturas de las alteraciones más comunes al realizar una capnografía.
Para más información del sistema respiratorio consultar las referencias[10 13 3]

3.3. Temperatura corporal

En condiciones normales, la temperatura corporal se encuentra entre 35,8 y 37,2 °C, con variaciones durante el día que hacen que esta temperatura sea más elevada por la tarde.
El hipotálamo es el responsable de la regulación de la temperatura corporal usando los siguientes mecanismos:

- **El sudor:** cuando la temperatura es elevada, las glándulas sudoríparas producen sudor y este se evapora en la superficie del cuerpo eliminando el calor.
- **La circulación cutánea:** cuando la temperatura es elevada las arterias cutáneas se dilatan, permitiendo que llegue mayor cantidad de sangre a la superficie de la piel y haciendo con ello que se enfríe, sin embargo, cuando la temperatura es baja, realiza el proceso inverso.
- **Contracción muscular:** el frío produce contracciones musculares involuntarias (temblores), que aumentan el tono muscular y consumen energía que se transforma en calor.
- **Piloerección:** el pelo cutáneo se eriza a la contracción de unos pequeños músculos que hay en la base de cada pelo, esto produce lo que comúnmente se conoce como "piel de gallina". En los humanos al carecer de una gran cantidad de pelo su efecto es nulo, pero en especies con pelo tupido permite que quede atrapada una capa de aire debajo del pelo aislando y disminuyendo con ello la pérdida de calor.
- **Aumento del metabolismo**: se incrementa la secreción de hormonas en la glándula tiroides, estas estimulan la producción de calor en todas las células del organismo.

3.3.1. Patologías por incremento de la temperatura corporal

Existen dos situaciones caracterizadas por aumento de temperatura corporal que conviene diferenciar:

- **Fiebre:** es el aumento temporal en la temperatura del cuerpo en respuesta a alguna enfermedad o padecimiento. Hablamos de febrícula si la temperatura está por encima de lo normal pero es menor de 38º C y de fiebre si se superan los 38 °C
- **Hipertermia:** producido por el mal funcionamiento del centro termorregulador, con temperaturas iguales o superiores a 41 °C. El aumento descontrolado de la temperatura origina importantes lesiones orgánicas, por lo que la hipertermia implica un importante riesgo para la salud, de ahí la importancia de un diagnóstico y tratamiento temprano.

3.3.2. Patologías por descenso de la temperatura corporal

- **Hipotermia:** es el descenso involuntario de la temperatura corporal (por debajo de 35ºC). El corazón, el sistema nervioso y otros órganos no pueden funcionar normalmente. Si no se trata a tiempo, puede provocar una parada cardiorespiratoria y finalmente, la muerte.

3.4. Actividad electrodérmica (EDA) y conductancia de la piel

En la lectura de la actividad electrodérmica (**EDA**) se monitoriza las características eléctricas de la piel, por ejemplo, la conductancia, causada por la variación de la sudoración del cuerpo humano.
La respuesta eléctrica está relacionada con la activación del sistema nervioso, con ello podemos detectar cambios emocionales, como un estado de ansiedad o de estrés, estas situaciones provocan un aumento de la sudoración en las manos y esto refleja una mayor conductividad de la piel.
Para obtener una señal EDA se colocan electrodos en los dedos índice y corazón de una mano. Como medida de la actividad electrodérmica se usa la la variación de una corriente de bajo voltaje aplicada entre los dos electrodos.

La monitorización de la actividad electrodérmica también se ha realizado para otros fines distintos como el polígrafo de la verdad, medir el esfuerzo psicofisiológico de los pilotos del ejercito del aire, actividades de investigación en el campo de la neurociencia, estudio en varios campos de la psicología y mucho más.

Capítulo 4

Entorno de desarrollo

4.1. Entorno Hardware

Para del desarrollo del proyecto se ha hecho uso de dos sistemas empotrados de la familia Arduino, estos son; **Arduino UNO** y **Arduino Nano** para la conexión con los diferentes sensores y una **Raspberry pi 3 b+** para recoger la información y mostrarla a través de un monitor desarrollado en Java.
La elección del uso de Arduino como plataforma para la conexión de los sensores, se debe a su respuesta inmediata, este dispone de un microcontrolador que ejecuta cíclicamente una rutina previamente cargada en memoria sin necesidad de un sistema operativo, lo que lo convierte en una opción idónea para obtener resultados en tiempo real. Para mostrar la información a través de una interfaz gráfica en un monitor, necesitamos un ordenador que además sea compacto, por ello se ha elegido la Raspberry al tratarse de un mini ordenador de bajo consumo y bajo coste.

4.1.1. Arduino Uno

Es el más popular de la familia Arduino, idóneo para iniciarse en la programación de sistemas empotrados y muy usado en el ámbito educativo. Además, se trata de una placa de hardware libre, lleva el microcontrolador **Atmega328**, dispone de alimentación externa y se conecta con un cable usb tipo A-B.

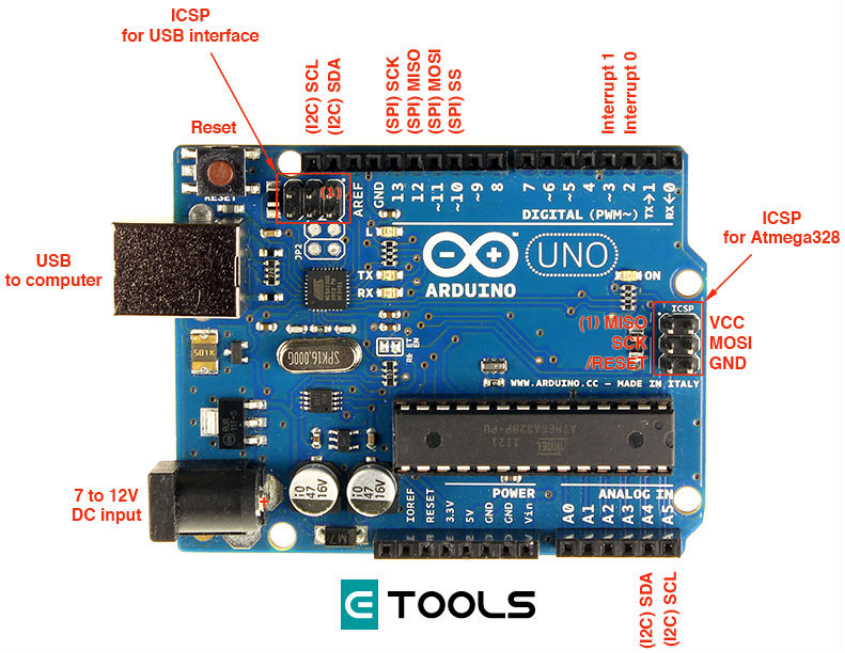


Figura 4.1: *Arduino Uno y datasheet (Electrontools.com[18])*

Características

- **Microcontrolador:** ATmega328
- **Voltaje Operativo:** 5v
- **Voltaje de Entrada (Recomendado):** 7 – 12 v
- **Pines de Entradas/Salidas Digital:** 14 (De las cuales 6 son salidas PWM)
- **Pines de Entradas Análogas:** 6
- **Memoria Flash:** 32 KB (ATmega328) con 0,5 KB destinados al Bootloader.
- **SRAM:** 2 KB (ATmega328)

- **EEPROM:** 1 KB (ATmega328)
- **Velocidad del Reloj:** 16 MHZ.
- **Dimensiones:** 2,1"x 2,3"
- **Longitud:** 68,6 mm
- **Ancho:** 53,3 mm
- **Peso:** 24 g

4.1.2. Arduino Nano

El Arduino Nano es una pequeña placa con un microprocesador ATmega168 . En funcionalidad es muy parecido al Arduino UNO pues tiene el mismo número de entradas y salidas disponibles pero más compactas, No posee conector para alimentación externa y funciona con un cable USB Mini-B en vez del cable estándar.

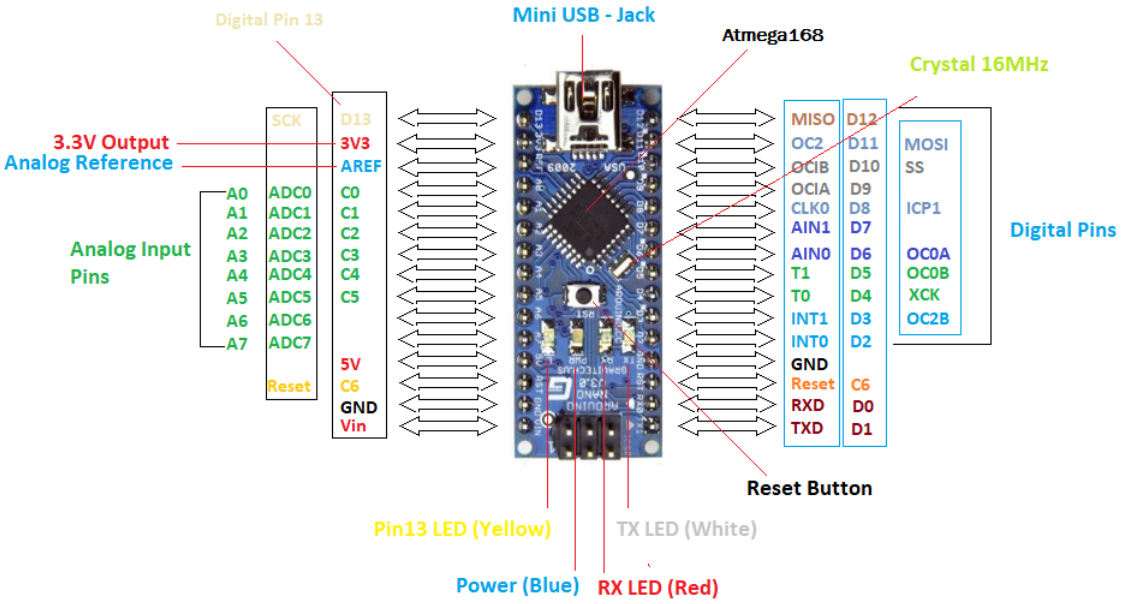


Figura 4.2: *Arduino Nano y datasheet (Theengineeringprojects.com[5])*

Características

- **Microcontrolador:** Atmel ATmega168
- **Tensión de funcionamiento (nivel lógico):** 5 V
- **Voltaje de entrada (recomendado):** 7-12 V
- **Voltaje de entrada (límites):** 6-20 V
- **E / S digitales:** 14 (de los cuales 6 proporcionan salida PWM)
- **Pines de entrada analógica:** 8

- **Corriente continua para Pin I / O:** 40 mA
- **Memoria flash:** 16 KB ATmega168 de los cuales 2 KB utilizado por el gestor de arranque
- **SRAM:** 1 KB (ATmega168)
- **EEPROM:** 512 bytes (ATmega168)
- **Velocidad de reloj:** 16 MHz
- **Dimensiones:** 0,73 "x 1,70"
- **Longitud:** 45 mm
- **Ancho:** 18 mm
- **Peso:** 5 g

4.1.3. Raspberry Pi 3 b+

La **Raspberry Pi 3 b+** es una placa computadora de bajo coste y reducidas dimensiones fabricada por la *Fundación Raspberry Pi*, posee los componentes principales que llevaría cualquier ordenador y dispone de su propio sistema operativo oficial basado en Debian llamado **Raspbian**. Es muy popular en el ámbito educativo por su versatilidad y características, tiene una multitud de pines **GPIO** (*General Purpose Input/Output*), tratándose de una interfaz más que aumenta las capacidades para crear diversos proyectos que podremos encontrar en su comunidad de creadores.

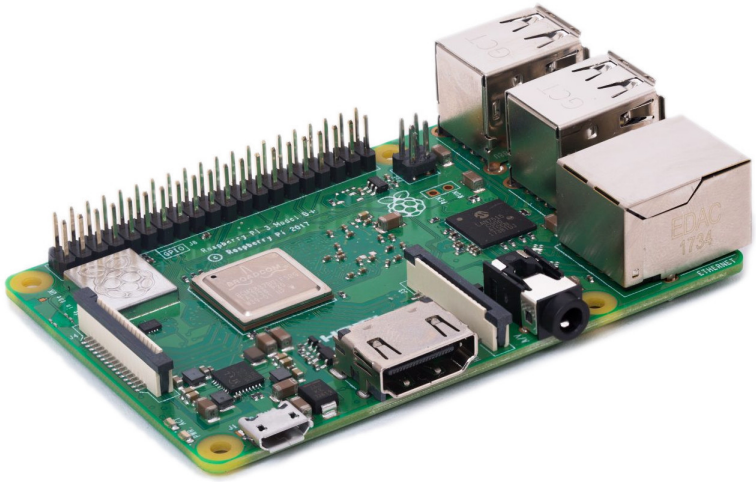


Figura 4.3: *Raspberry Pi 3 b+ (Raspberrypi.org[4])*

Características

- CPU + GPU: Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- RAM: 1GB LPDDR2 SDRAM
- Wi-Fi + Bluetooth: 2.4GHz y 5GHz IEEE 802.11.b/g/n/ac, Bluetooth 4.2, BLE
- Ethernet: Gigabit Ethernet sobre USB 2.0 (300 Mbps)
- GPIO de 40 pines
- HDMI
- 4 puertos USB 2.0

- Puerto CSI y DSI para conectar una cámara y una pantalla táctil
- Salida de audio estéreo y vídeo compuesto
- Micro-SD
- Power-over-Ethernet (PoE)

4.1.4. Monitor de frecuencia cardíaca AD8232

La placa **SparkFun AD8232 monitor de frecuencia cardíaca**, se utiliza para medir la actividad eléctrica del corazón. Esta actividad eléctrica se puede graficar como un **ECG** o un **electrocardiograma** y emitirse como una lectura analógica.
Está diseñado para extraer, amplificar y filtrar pequeñas señales biopotenciales en presencia de condiciones ruidosas, como las creadas por el movimiento o la colocación remota de electrodos, y así, ayudar a obtener una señal clara de los intervalos de **PR** y **QT** fácilmente.

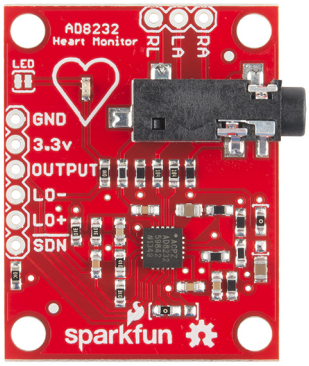


Figura 4.4: *Circuito AD8232 (Sparkfun[1])*

El monitor de ritmo cardíaco **AD8232** dispone de nueve conexiones en el circuito integrado a las que puede soldar pines, cables y otros conectores, también dispone de un conector jack de 3.5mm que permite conectar los electrodos que captarán las señales y un LED que parpadea con el ritmo de un latido del corazón.

Conexiones del lateral izquierdo de la figura 4.4.

- **GND:** Toma de tierra.
- **3.3V:** Fuente de alimentación.
- **OUTPUT:** Señal de salida.
- **LO-:** Detección de derivaciones -
- **LO+:** Detección de derivaciones +

- **SDN:** Desconectar la placa.

En la parte superior 4.4 están las conexiones:

- **RA (Right Arm):** Conexión del brazo derecho.
- **LA (Left Arm):** Conexión del brazo izquierdo.
- **RL (Right Leg)** Conexión de la pierna derecha.

4.1.5. TechPatient CARDIO ECG Simulator

TechPatient cardio es un simulador ECG portátil que reproduce patrones de onda de un sujeto humano sano para generar nuevas ondas cardíacas realistas, con control de la amplitud, la frecuencia y la variabilidad latido a latido.

Figura 4.5: *Simulador ECG (He Instruments[2])*

Las posibilidades de ensayo que permite son de 12, 5 y 3 derivaciones. Pueden seleccionarse diferentes modos de onda con parámetros específicos para cada modo.

El **Modo ECG** produce ondas cardíacas realistas en 12 derivaciones parametrizadas en amplitud y frecuencia cardíaca.

El **Ritmic Module** simula arritmias, dispone de 45 arritmias prefijadas con amplitud configurable.

Las **Ondas de Performance** están diseñadas para ensayo y medición. Posibilidad de definir parámetros de amplitud, frecuencia y nivel.

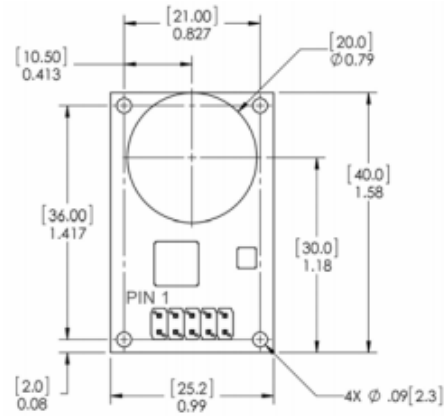
4.1.6. ExplorIR-W CO_2 Sensor

El sensor de dióxido de carbono ExplorIR-W brinda mediciones de CO_2 a 20Hz, 10 veces la velocidad de un sensor NDIR (NonDispersive InfraRed) normal de 2Hz, es el sensor de CO_2 más preciso del mercado.
Dispone de un microcontrolador que permite la comunicación con la placa a través de comandos en Serial UART (normally high) a 9600 baudios.

Figura 4.6: *Sensor de CO_2 (co2meters.com[16])*

La comunicación en serie a través de comandos[17] nos permite definir parámetros de configuración u obtener las lecturas del sensor.

A continuación se muestra la función y orden de cada pin de conexión en la placa, para más información consultar datasheet[16].



Function	Pin #	Pin #	Function
0V	1	2	N/C
+3.`V	3	4	0V
Sensor Rx (in)	5	6	0V
Sensor Tx (out)	7	8	Zero N
N/C	9	10	Zero Air

[1]

[1]Esquema de pinout obtenido en datasheet[17]

4.1.7. Sonda de temperatura DS18b20

El DS18B20 puede medir temperaturas entre -55ºC y 125ºC, dado que es un sensor digital, la señal leída no se degrada debido a la distancia del cableado.Incorpora una memoria de 64-bit para almacenar el identificador o dirección única de cada sensor, esta dirección es necesaria dentro del bus 1-Wire para identificar cada uno de los sensores de temperatura DS18B20 conectados al bus de comunicación, esto permite dos cosas, por un lado robustez en la transmisión de los datos ya que trabaja con datos digitales, mucho menos sensibles a los efectos adversos del ruido que las señales analógicas y por otro lado, permite conectar muchos sensores de temperatura con un único pin digital. Ver datasheet[14].

Figura 4.7: *Patillaje ds18b20 (Programafacil.com[12])*

Características técnicas del DS18B20

- Rango de temperatura: -55 a 125ºC
- Resolución: de 9 a 12 bits (configurable)
- Interfaz 1-Wire (Puede funcionar con un solo pin)
- Identificador interno único de 64 bits
- Múltiples sensores puede compartir el mismo pin
- Precisión: ±0.5ºC (de -10ºC a +85ºC)
- Tiempo de captura inferior a 750ms
- Alimentación: 3.0V a 5.5V

4.2. Entorno Software

4.2.1. Arduino IDE

El entorno de desarrollo integrado (IDE) de Arduino disponible en las principales plataformas proporciona las herramientas para el desarrollo de rutinas en Arduino y también para la comunicación bidireccional con las placas.

Figura 4.8: *Arduino IDE.*

Con este entorno de desarrollo se ha realizado la programación de todas las rutinas para las placas, con ello tenemos a disposición herramientas de validación del código, carga de los programas en memoria, disponibilidad de una gran variedad de Librerías con sus ejemplos de uso y dos tipos de monitores del puerto serie, un monitor bidireccional donde podemos enviar datos a la placa o mostrar los datos recibidos por la misma 5.9 y otro que realiza una gráfica en tiempo real de los datos que se recibe de la placa 5.3.

4.2.2. Fritzing

Fritzing es un programa de software libre disponible en las principales plataformas, con la finalidad de automatizar el diseño electrónico, esta herramienta ayuda a los diseñadores a pasar de prototipos a productos finales. También permite documentar prototipos basados en Arduino y crear esquemas de circuitos impresos para su posterior fabricación.

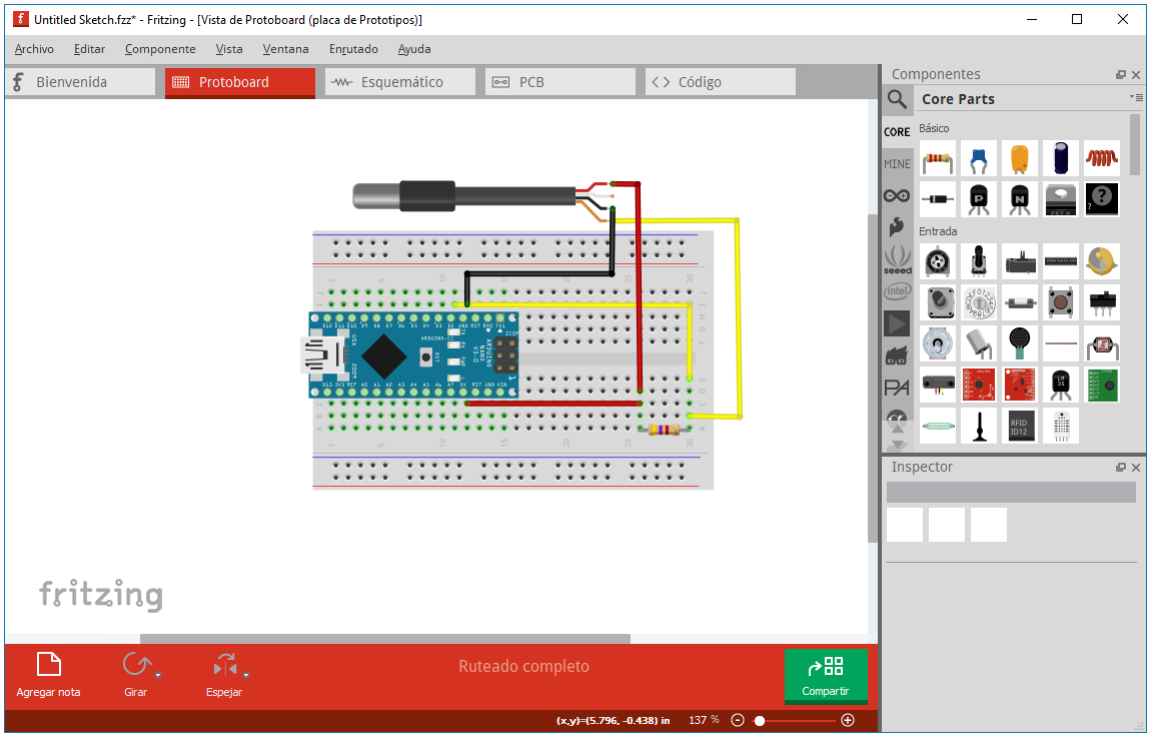


Figura 4.9: *Fritzing con esquema de sensor de temperatura.*

En el desarrollo de este proyecto se ha utilizado esta herramienta para el diseño de los esquemas de conexión de los sensores con las placas, ya que dispone de multitud de componentes usualmente utilizados en Arduino y sus propias placas, lo que nos ha ofrecido una buena solución para realizar esta tarea.

4.2.3. Processing IDE

Processing IDE es un entorno de desarrollo con capacidad para programar en diferentes lenguajes, pero principalmente en su propio lenguaje basado en Java. Sus herramientas nos permite aplicaciones gráficas siguiendo una estructura predefinida en la que cíclicamente ejecuta la función *draw()* que dibuja los componentes.

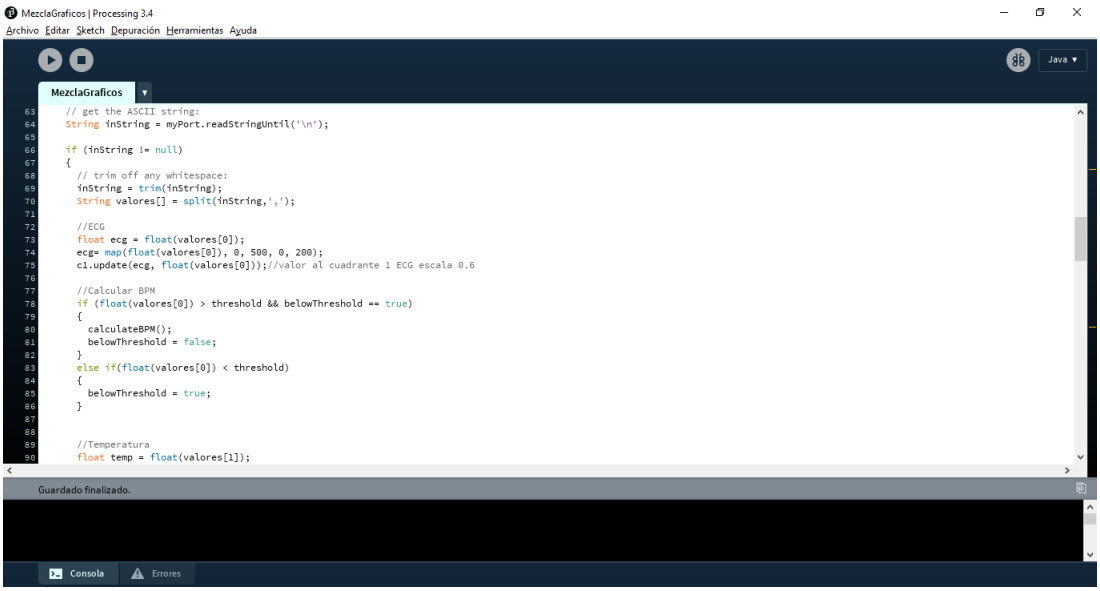


Figura 4.10: *Processing IDE*

Este entorno de desarrollo se ha empleado para desarrollar la interfaz gráfica del monitor, donde se muestra la información de todos los sensores. Se ha escogido esta opción por su facilidad de aprendizaje y su uso del lenguaje Java.

Capítulo 5

Desarrollo

Una vez determinada la temática, se marcó un plan de trabajo a seguir 1.3, después, se fue trabajando y documentando cada uno de los objetivos marcados 1.2, donde primero se estudian los dispositivos a utilizar, se realiza un análisis de la materia que se abarca, se investiga como funciona cada placa o sensor y se hace el desarrollo y documentación de cada hito. En último lugar, se unifican los desarrollos a una sola placa, se crea la interfaz gráfica y se documenta el proceso al completo.

5.1. Desarrollo de electrocardiograma

Para el uso del electrocardiograma únicamente usamos los pines esenciales, GND, VCC (3,3V) y salida de datos (Output). Para obtener las señales, usamos el puerto jack de 3.5mm con tres electrodos y la lectura de resultados se puede realizar tanto en el simulador de paciente como en una persona.

5.1.1. Componentes

Los materiales usados para este proyecto son:

- Arduino UNO 4.1.
- Placa AD8232. 4.4.
- Electrodos con conexión jack 3.5mm.
- Simulador de paciente *Tech Cardio* 4.5.

- Adhesivos corporales (En pruebas reales).
- Cables.

5.1.2. Esquema de montaje

En este esquema se detalla como realizar las conexiones, tanto en Arduino, como en una persona.

Arduino	AD8232
3.3V	3.3V
GND	GND
A0	OUTPUT

AD8232	Señal
RA (Negro)	Brazo derecho
LA (Azul)	Brazo izquierdo
RL (Rojo)	Pierna derecha

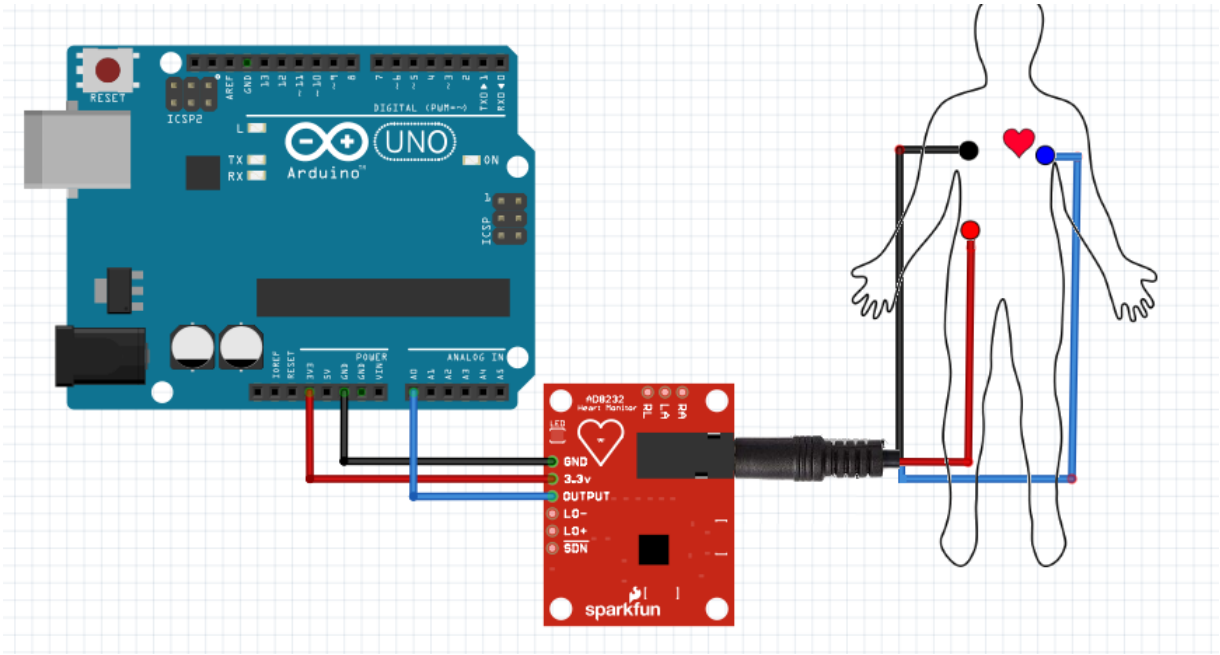


Figura 5.1: *Esquema conexiones electrocardiograma*

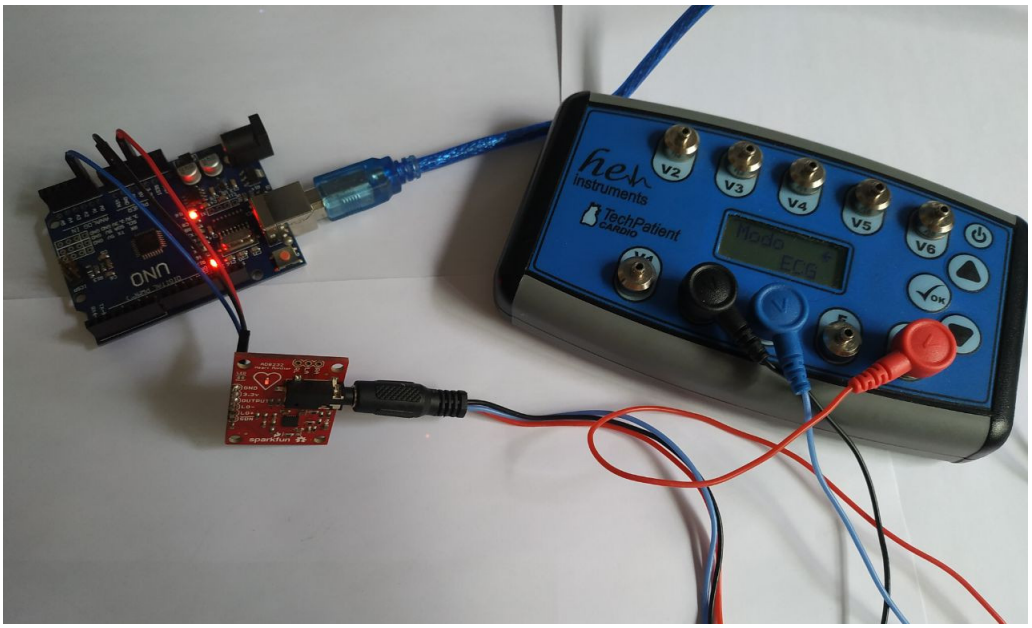


Figura 5.2: *Conexiones en simulador de paciente.*

5.1.3. Resultados

Una vez realizadas las conexiones y cargado en memoria el programa, la información se envía por el puerto serie al ordenador y éste muestra los resultados.

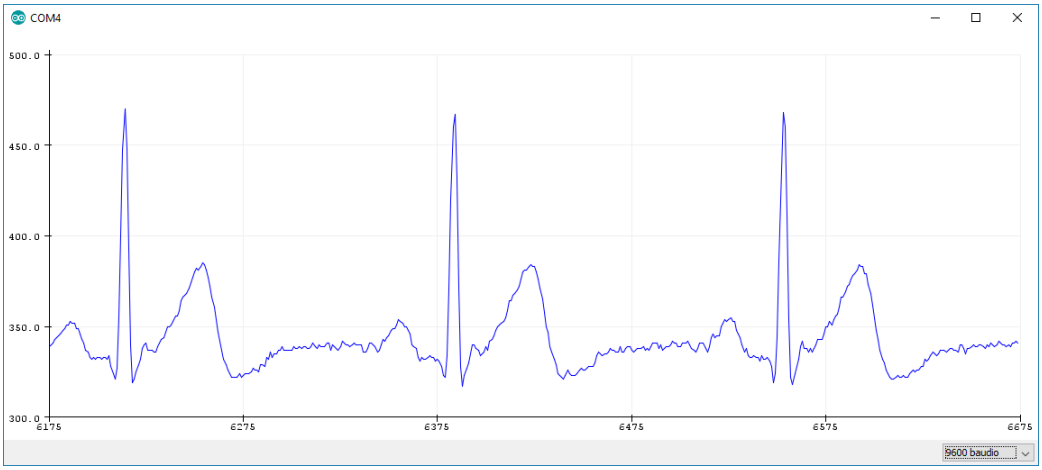


Figura 5.3: *Monitor electrocardiograma*

5.2. Desarrollo de monitor de CO2

Para las pruebas de medición de CO_2 se hará uso del sensor ExplorIR-W CO_2 4.1.6, con el que es necesario mantener una conexión bidireccional para enviar comandos al sensor y recibir la información solicitada. Para ello se conecta a una placa Arduino a través de los pines de transmisión, lo que permitirá la comunicación con el puerto Serial de Arduino. Para el desarrollo final con la unificación de todos los sensores, se usará un puerto Serial virtual exclusivo con este sensor, ya que de lo contrario saturaría el canal.

5.2.1. Componentes

Los materiales usados para este proyecto son:

- Arduino Uno 4.1.
- ExplorIR-W CO_2 Sensor 4.6.
- Cables.

5.2.2. Esquema de montaje

Para conectar el sensor ExplorIR-W con Arduino realizamos las siguientes conexiones, con ello podemos comunicar la placa con el sensor a través de comandos por el UART standard.

Arduino	**ExplorIR-W**
GND	Pin 1 GND
3.3V	Pin 3 3.3v
TX (D1)	Pin 5 RX (IN)
RX (D0)	Pin 7 TX (OUT)

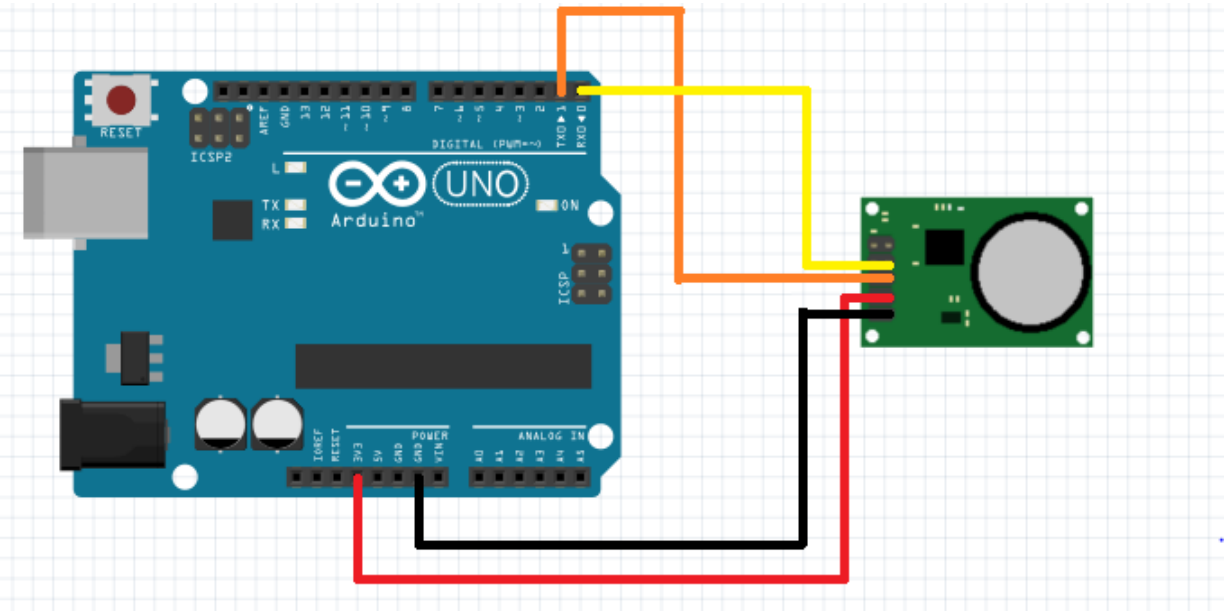


Figura 5.4: *Esquema conexiones para sensor de CO_2.*

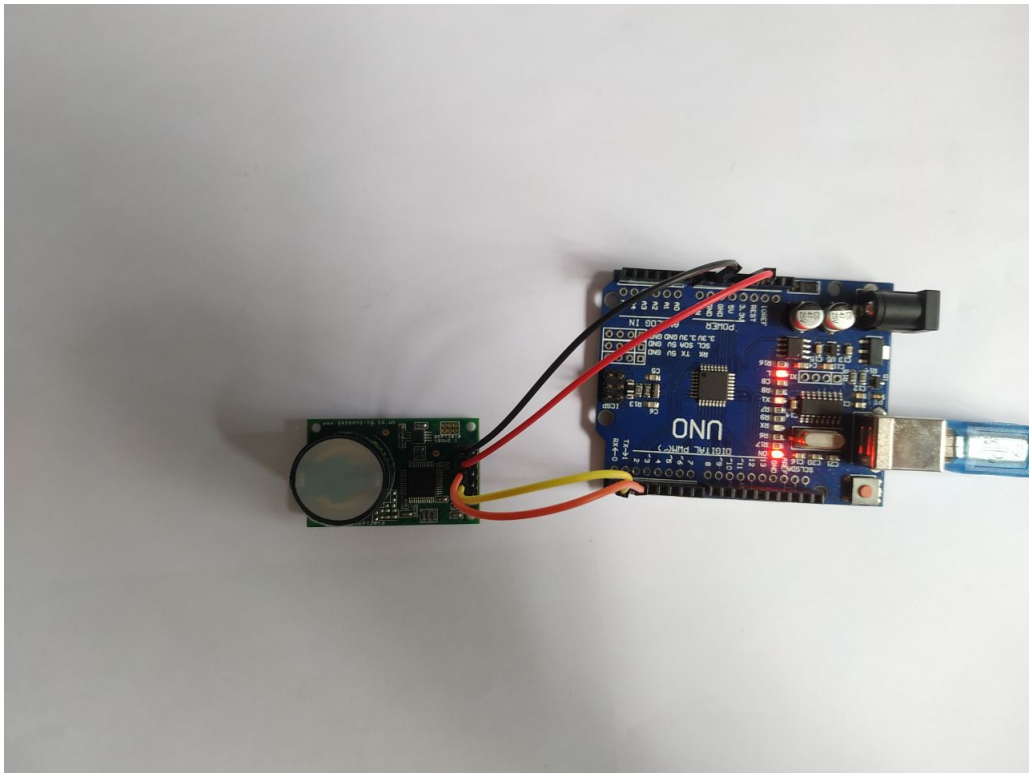


Figura 5.5: *Conexiones para sensor de CO2*

5.2.3. Resultados

Antes de realizar las conexiones debemos cargar primero la rutina en la memoria del Arduino, ya que de lo contrario, al tener conectado el sensor en los pines de transmisión, no se cargaría en la memoria. Cuando ya tengamos la rutina cargada realizamos las conexiones.

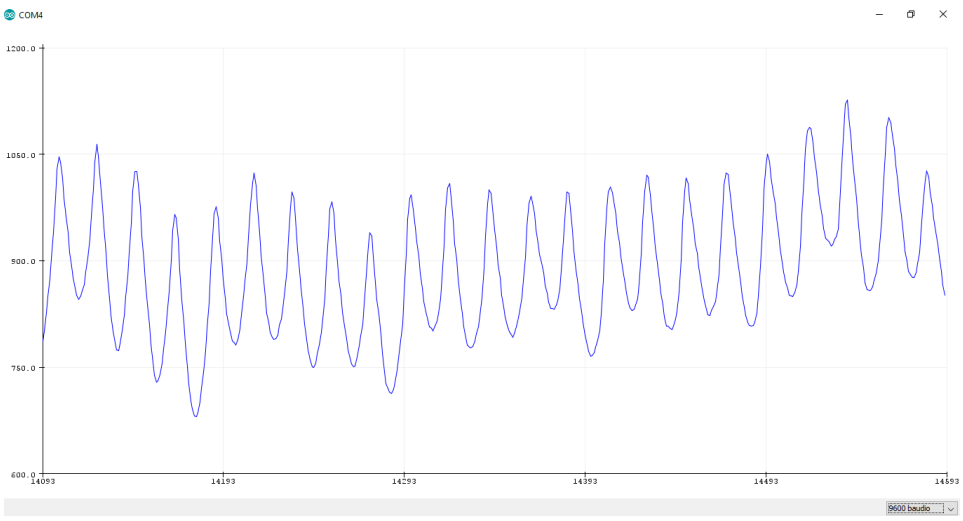


Figura 5.6: *Monitor CO_2 en aire exhalado.*

5.3. Desarrollo de sensor de temperatura

Para el desarrollo del sensor de temperatura, en primer lugar se usó un termistor ntc de uso médico, el problema de éste, es que se desconocía el fabricante, por lo tanto no se encontró documentación. Finalmente, tras varios intentos fallidos para obtener resultados, se sustituyo por la sonda ds18b20, con la que finalmente se logró el objetivo.

5.3.1. Componentes

Los materiales usados para este proyecto son:

- Arduino Nano 4.2.
- Sonda DS18B20 4.7.
- Resistencia de 4.7KΩ.
- Protoboard.
- Cables.

5.3.2. Esquema de montaje

La sonda tiene 3 cables de conexión, el rojo de VCC, el negro de GND y el amarillo de datos. Conectamos VCC a 5V, GND a uno de las conexiones GND de arduino y el cable de datos a una de las entradas digital, en este caso usaremos la número 8, por último, conectamos la resistencia Pull-UP de 4.7K entre la conexión de datos y VCC.

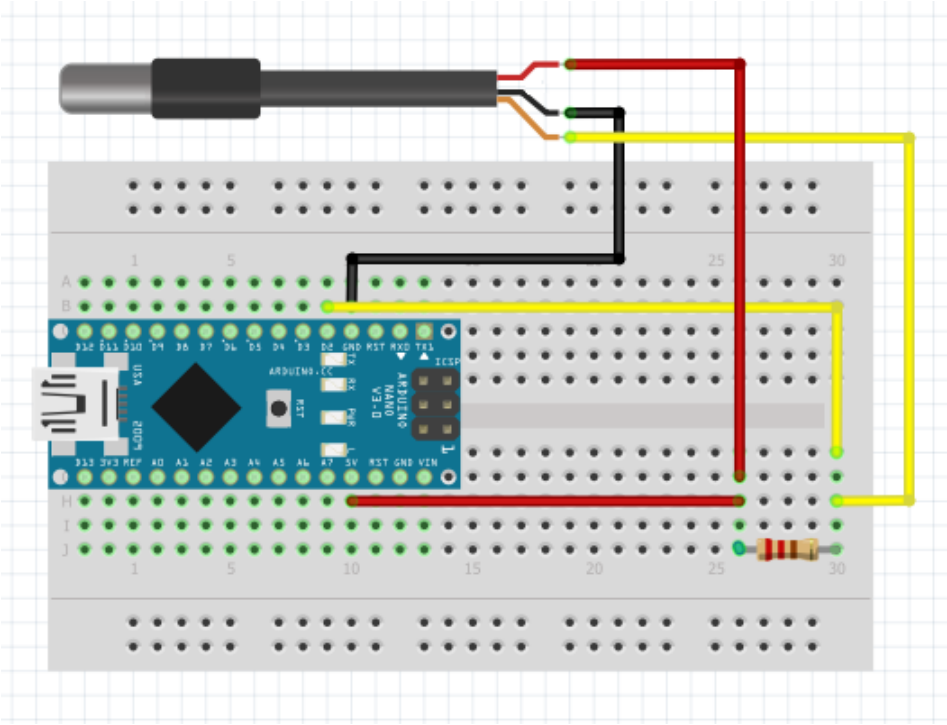


Figura 5.7: *Esquema conexiones para sensor de temperatura*

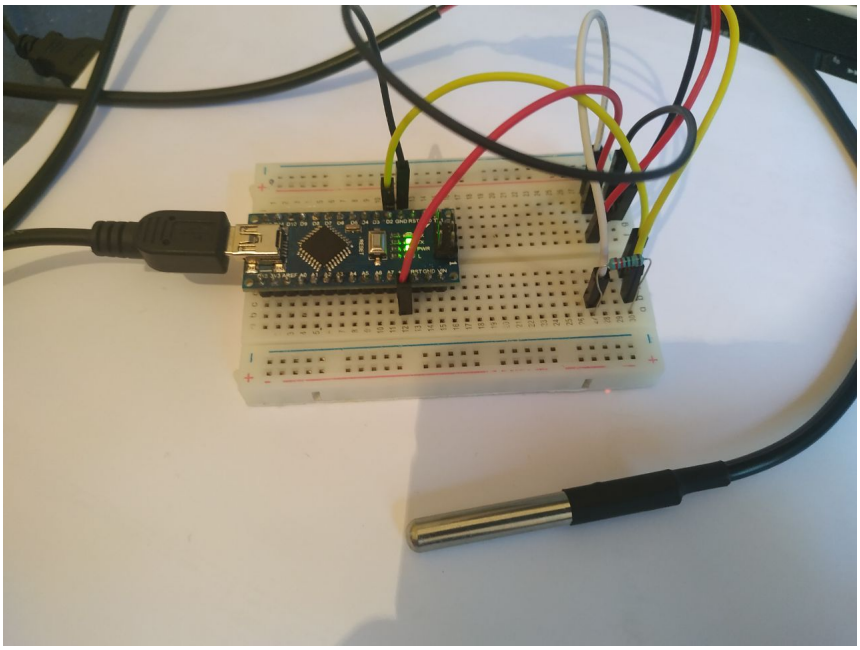


Figura 5.8: *Conexiones para sensor de temperatura*

5.3.3. Librerías

Para programar el uso del *DS18b20* en Arduino se han utilizado dos librerías:

- Librería **OneWire**: el *DS18b20* hace usa el protocolo del bus one-wire, por lo tanto hacemos uso de esta librería diseñada para el uso de este protocolo.
- Librería **DallasTemperature:** necesaria para realizar las lecturas o configuraciones del *DS18b20*.

5.3.4. Resultados

Una vez realizadas las conexiones y cargado en memoria el programa, la información se envía por el puerto serie al ordenador, donde podremos mostrar las lecturas de temperatura.

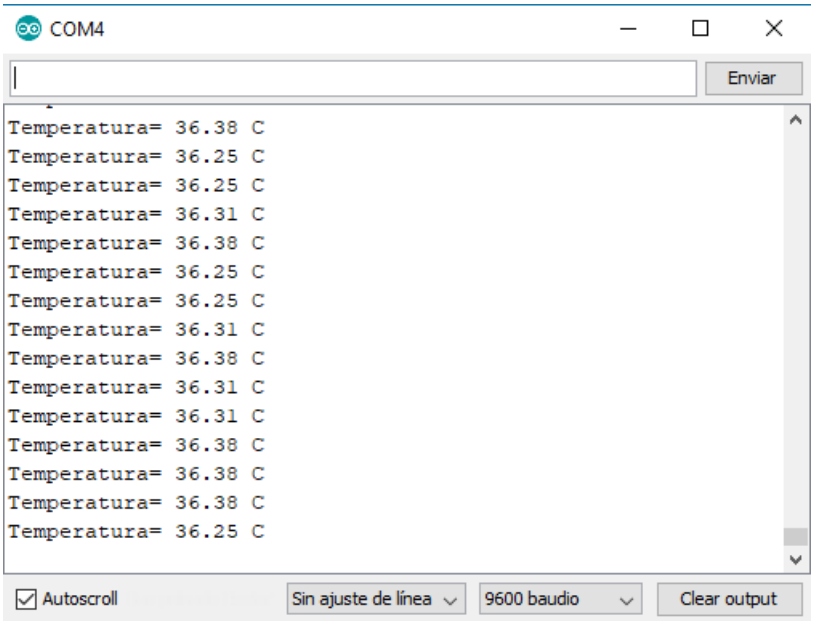


Figura 5.9: *Lectura de temperatura corporal*

5.4. Desarrollo de sensor de actividad electrodérmica (EDA)

Usando la ley de voltaje de Kirchhoff en mallas, con electrodos conectados a dos dedos de la mano y una resistencia, se podrían detectar cambios en la conductividad debido a la sudoración de los electrodos conectados a los dedos. (Ver figura 5.11).

5.4.1. Componentes

Los materiales usados para este proyecto son:

- Arduino Nano 4.2.
- Dos electrodos con velcro.
- Resistencia de 10KΩ.
- Protoboard.
- Cables.

5.4.2. Esquema de montaje

Uno de los electrodos se conecta a 5V de Arduino, en una de las patillas de la resistencia se conecta GND, en la otra se conecta el otro electrodo y la salida de lectura analógica que irá conectado al pin A0 analógico de Arduino y por último, se coloca un electrodo a cada dedo.

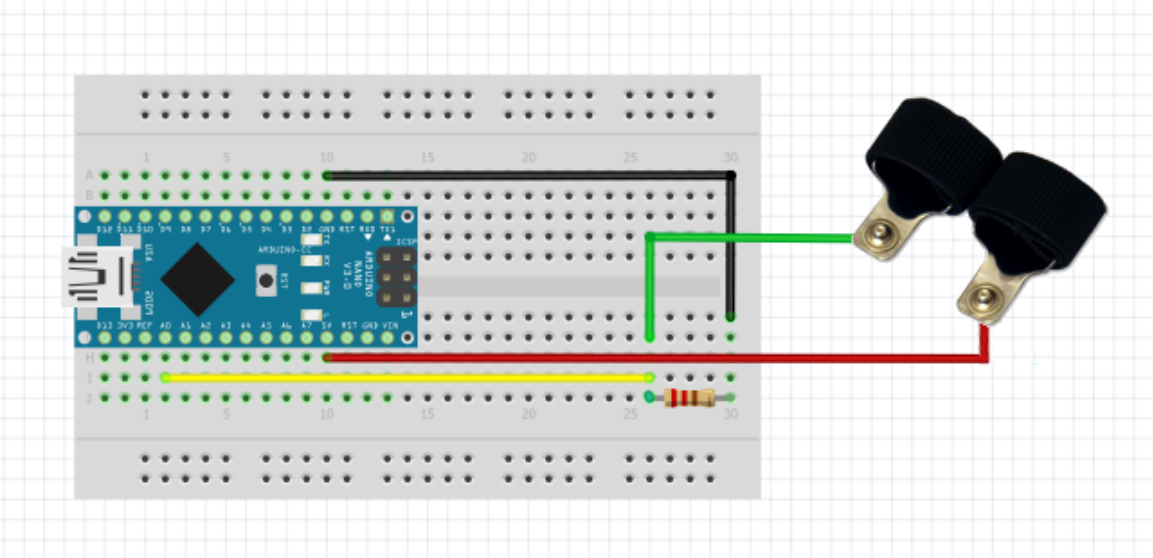


Figura 5.10: *Esquema conexiones para sensor de EDA*

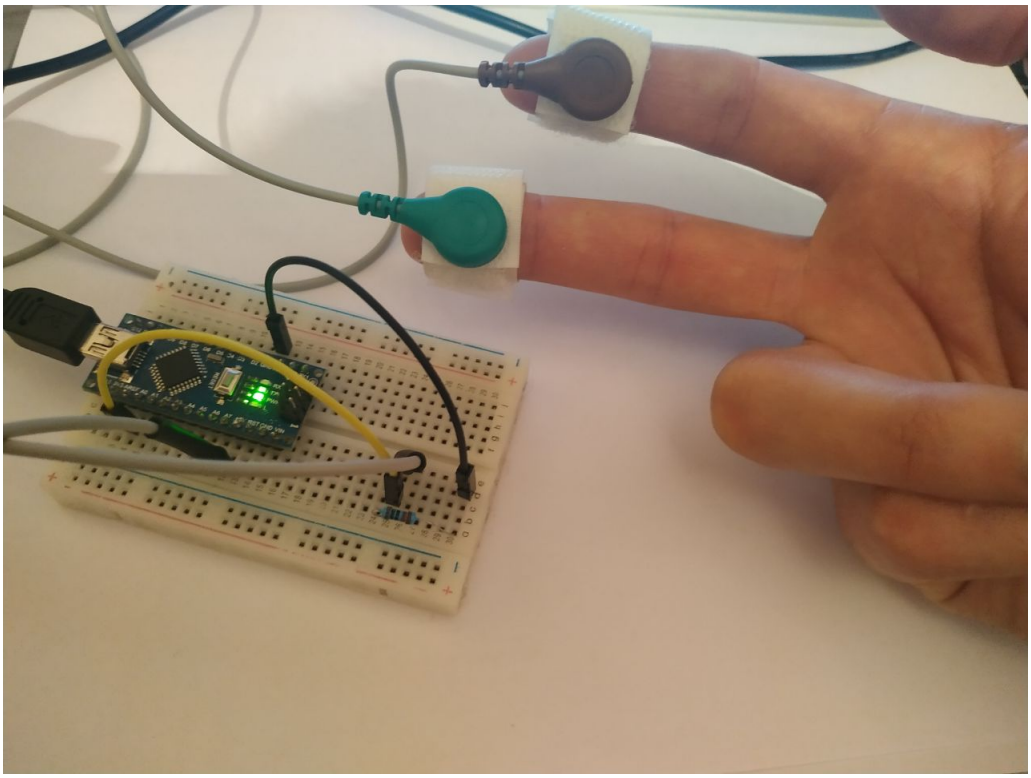


Figura 5.11: *Conexiones para sensor EDA*

5.4.3. Resultados

Una vez realizadas las conexiones y cargado en memoria el programa, la información se envía por el puerto serie al ordenador, donde podemos mostrarla.

Figura 5.12: *Simulación de cambio en los niveles de sudoración*

En esta prueba se observa el inicio de conexión, cuando se colocan los dedos en los electrodos, se realiza un muestreo en el que se mantiene estable en un valor, después de esto, se humedecen los dedos simulando la sudoración, por ello se ve otro valle a cero, momento de desconexión y después una lectura con valores mayores, esto se debe a que ahora hay mayor conductividad.

5.5. Acoplar los sensores en una sola placa

Para unificar todo lo anterior se han tenido que realizar varios cambios, tanto en las conexiones, como en la estructura del código, lo que ha supuesto varios problemas a solucionar.
Uno principal es la ejecución monoproceso, el contener una única rutina secuencial supone un problema para la comunicación con todos los sensores, dado que cada uno precisa de diferentes tiempos de espera, esto genera un retardo mayor por la espera de todos los componentes y este retardo es especialmente relevante para obtener buenos resultados del ECG. Como solución, el mayor tiempo de espera se intercala con las lecturas del sensor cardíaco evitando que afecte a su funcionamiento y simulando una ejecución multiproceso.

```
//Pseudocodigo de ejemplo
readAllSensors();
sendAllData();
//Cincuenta iteraciones con delay 1ms
for(int i=0; i<50; i++){
    readECG();
    sendAllData();
    delay(1);
}
```

Otro problema importante era que los pines *TX* y *RX* usados en el sensor *ExplorIR-W* 4.1.6 comparten la conexión del puerto serie por USB. Esto es un problema dado que ahora la comunicación no es exclusiva con este sensor, si no que también se necesita la comunicación en serie con la *Raspberry* para mostrar la información de todos los sensores y el canal queda bloqueado. Como solución se han usado los pines digitales 10 y 11 para simular un puerto serie vía software, para ello se ha utilizado la librería *SoftwareSerial* así disponemos de un canal para la comunicación con este sensor y otro canal para la comunicación con la *Raspberry*.
Un último problema menor es el envío de la información distinguiendo a quien pertenece cada dato, que se soluciona enviando todos los datos simultáneamente separados por comas.
Una vez desarrollado el nuevo programa a cargar en la memoria de la placa y solucionados los problemas descritos, ya podemos conectar todos los sensores siguiendo el siguiente esquema de conexión.

5.5.1. Esquema de montaje

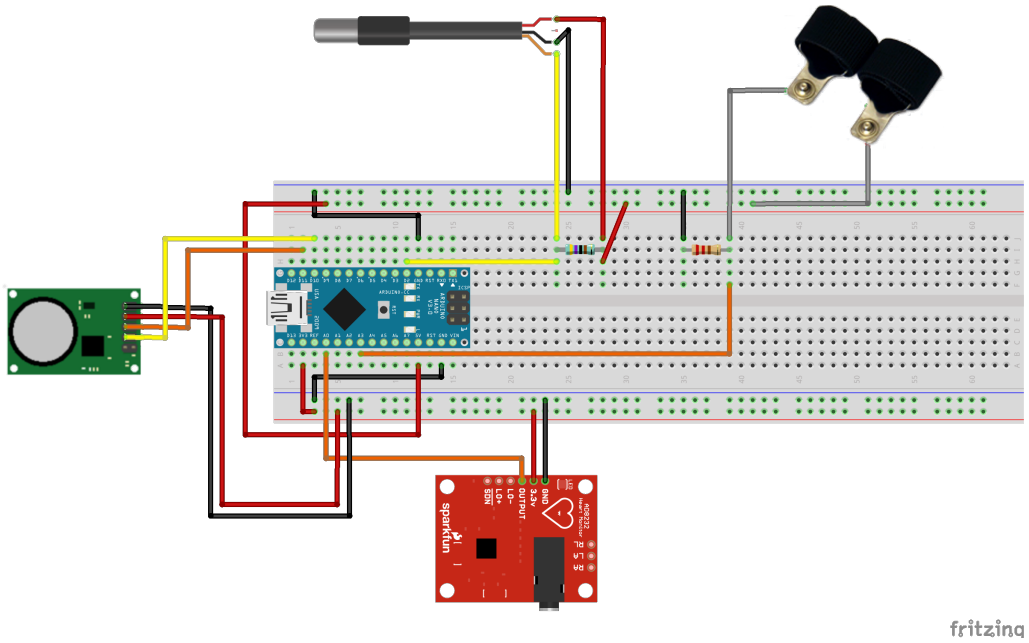


Figura 5.13: *Esquema general de conexiones con Arduino Nano.*

Con los sensores ExplorIR-W 4.1.6 y AD8232 4.1.4 se realiza una conexión directa sin necesidad de más componentes siguiendo las siguientes tablas.

Arduino	AD8232
3.3V	3.3V
GND	GND
A0	OUTPUT

Arduino	ExplorIR-W
GND	GND pin 1
3.3V	3.3V pin 3
D10 (STX)	RX pin 5
D11 (SRX)	TX pin 7

Para conectar la sonda ds18b20 4.1.7, se conecta el cable negro a GND, el cable rojo a 5V y el amarillo a pin D2 de Arduino, además, se debe añadir una resistencia de 4.7kΩ entre el cable de datos (amarillo) y el de tensión (rojo). Para la conexión de los electrodos destinados a la lectura de la actividad electrodérmica (EDA) 5.4, conectamos uno de los electrodos a 5V y para el otro se emplea una resistencia de 10kΩ, conectando una pata de la misma a GND y la otra al electrodo restante y al pin A3 de Arduino.

5.6. Desarrollo de interfaz gráfica

La información de todas las señales se obtiene en la placa Arduino y se envía a la Raspberry, donde se procesa y se muestran los datos haciendo uso de una aplicación desarrollada en Precessing basada en Java, se presenta una interfaz gráfica que consta de una sola ventana dividida en 4 secciones que muestra en cada una los datos de cada señal junto con un gráfico en tiempo real.

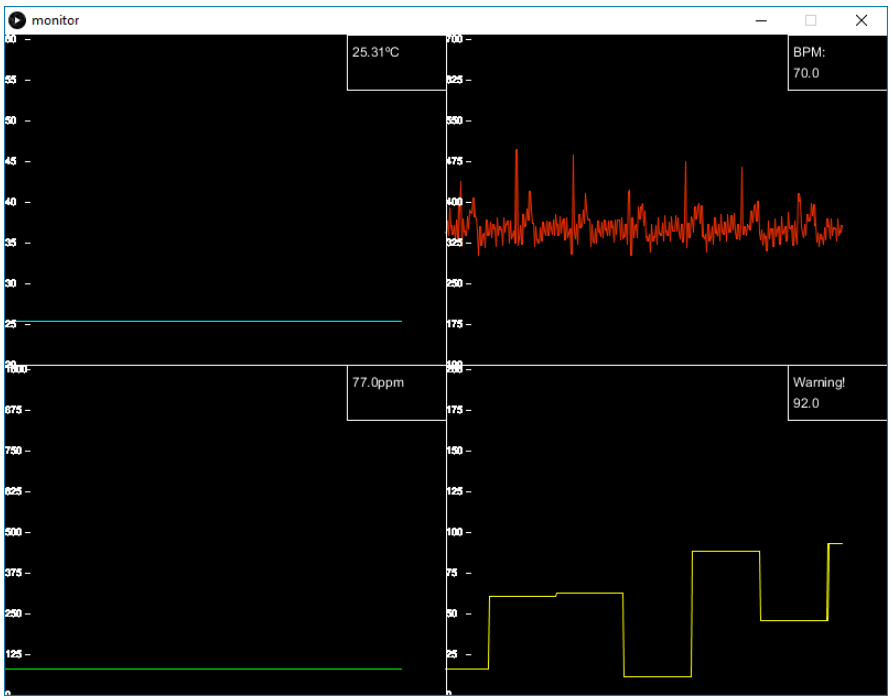


Figura 5.14: *Interfaz gráfica del monitor de señales.*

Para el diseño se dibujan figuras geométricas. La ventana forma un rectángulo con dos lineas divisorias a la mitad de cada eje, donde se crean patrones de dibujo para los cuadrantes, de forma que únicamente cambiando los valores de altura y anchura de la ventana, no se descuadre la imagen.
En cada cuadrante se crea un rectángulo en la esquina superior izquierda donde mostrar los datos numéricos correspondientes. Al realizar las gráficas se escalan los valores de las señales para poder mostrarlos en el espacio correspondiente. Los gráficos son lineas de puntos entre cada valor y para mantener la continuidad, una vez el éste llega al final, limpia la pantalla y continua en el principio.

Capítulo 6

Resultados y conclusiones

6.1. Resultados

Como resultado del trabajo se ha obtenido un monitor compacto que cumple con los objetivos marcados inicialmente. En la siguiente imagen podemos ver el resultado de la implementación del trabajo usando una tablet conectada a la Raspberry por VNC.

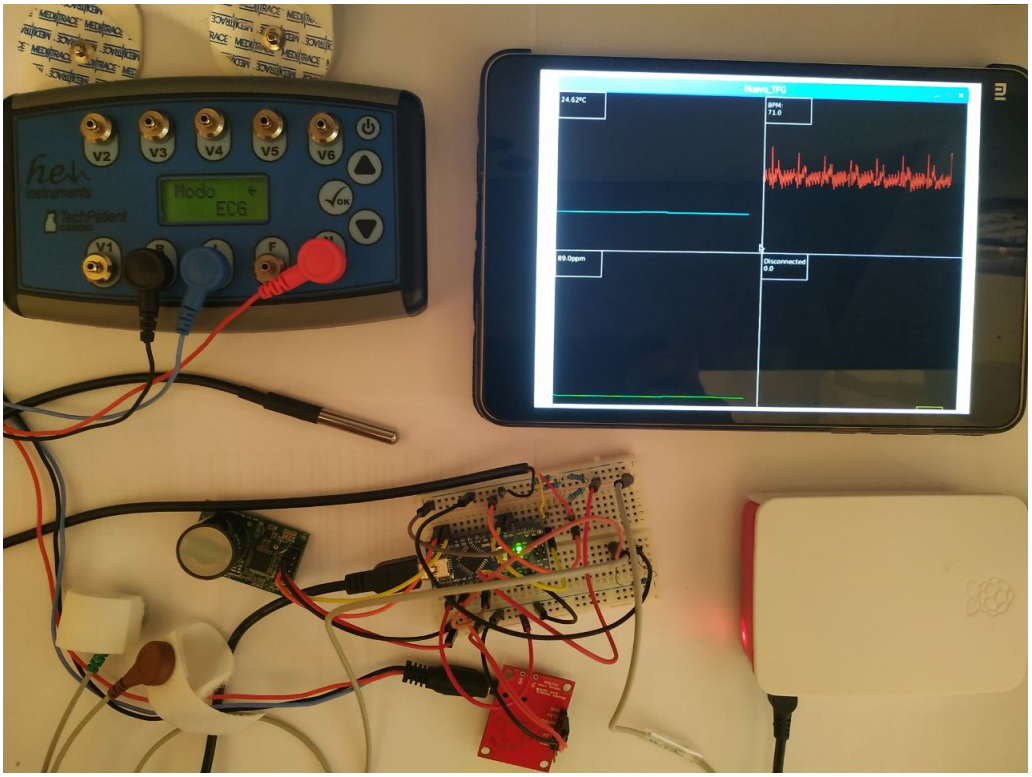


Figura 6.1: *Visión general del proyecto*

- El monitor es capaz de mostrar un gráfico del electrocardiograma en tiempo real, así como los BPM correspondientes.
- Puede medir los niveles de CO_2 en aire exhalado y mostrar un gráfico del mismo.
- Controla la temperatura corporal. En este caso, ante cambios bruscos, el tiempo de respuesta es mayor, pero mantiene lecturas estables.
- Permite monitorizar la actividad electrodérmica. Además del gráfico, indica si los electrodos están conectados, si se encuentra en valores normales y una alerta en caso de detectar valores de sudoración elevados.
- Finalmente se logra que todos los sensores estén acoplados a una sola placa de Arduino.
- Ejecución de interfaz gráfica a través de una Raspberry, lo que lo hace más compacto y permite conectarse a través de VNC o conectando un monitor.

Entre los posibles casos de uso aplicados a la realidad está, la monitorización de señales biomédicas con las que se puede detectar alguna posible anomalía o un estado de estrés en un individuo. Al tratarse de un dispositivo bastante compacto en comparación con los aparatos médicos profesionales, puede resultar una opción donde alguna anomalía pueda ocasionar un problema grave. Un ejemplo donde se usan dispositivos similares es en pilotos de aviones de combate del ejército español, y podría extrapolarse a otros ámbitos como la Fórmula 1.

6.2. Conclusiones

Una vez finalizado el desarrollo del proyecto llega el momento de realizar una valoración de lo conseguido y lo aprendido.

El objetivo de este proyecto era implementar diferentes dispositivos que obtengan señales biomédicas, ante lo que podemos afirmar que se ha logrado midiendo un total de cuatro señales. Además, se ha conseguido unificar la conexión de todos los sensores en una sola placa y se ha desarrollado una interfaz gráfica que facilita la visualización de los datos.

En consecuencia, se han ampliado los conocimientos sobre la materia de los sistemas empotrados así como sus capacidades y posibles usos en otros ámbitos.

Aunque se trate de un sistema funcional, no debe usarse para fines de diagnóstico médico ya que no se utiliza material profesional, si no de sensores que en su mayoría están pensados para otro propósito y han sido adaptados, es decir, no se garantiza fiabilidad en los resultados. Por ello y para que se trate de un sistema, además de funcional, también fiable, deberían usarse sensores profesionales desarrollados para este fin y que dispongan de una calidad garantizada.

En resumen y a pesar de las dificultades, se ha logrado realizar el proyecto planteado, lo que nos ha permitido ampliar y plasmar los conocimientos adquiridos en esta etapa haciéndonos sentir con ello muy satisfechos.

Capítulo 7

Results and conclusions

7.1. Results

As a result of the work, a compact monitor has been obtained that meets the objectives initially marked. In the following image we can see the result of the work implementation using a tablet connected to the Raspberry by VNC.

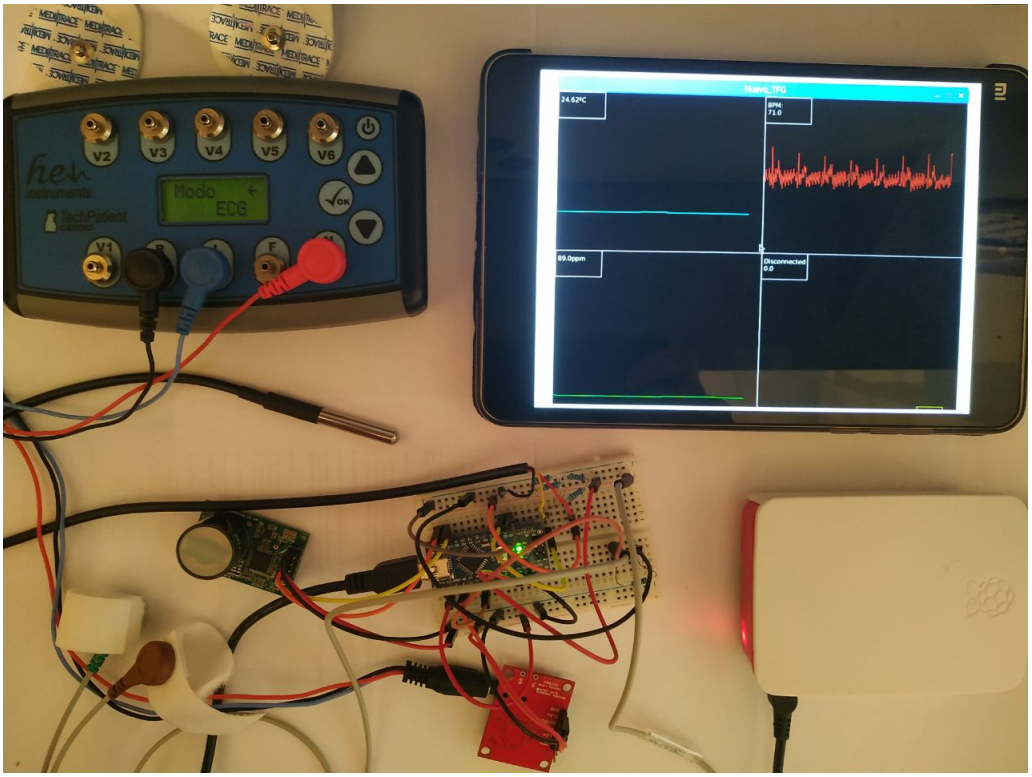


Figura 7.1: *Project overview*

- The monitor is able to show a graph of the electrocardiogram in real time as well as the corresponding BPM.
- You can measure the CO_2 levels in exhaled air and display a graph of it.
- Controls body temperature. In this case, in the face of sudden changes, the response time is longer, but it keeps stable readings.
- It allows to monitor the electrodermal activity. In addition to the graph it indicates if the electrodes are connected, if it is in normal values and an alert in case of detecting high sweat values.
- Finally, all the sensors are coupled to a single Arduino board.
- Graphical interface execution through a Raspberry which makes it more compact and allows to connect through VNC or connecting a monitor.

Among the possible cases of use applied to reality is the monitoring of biomedical signals with which a possible anomaly or a state of stress can be detected in an individual. As it is a fairly compact device compared to professional medical devices, it may be an option where an anomaly can cause a serious problem. An example where similar devices are used is in fighter pilots of the Spanish army, and could be extrapolated to other areas such as Formula 1.

7.2. Conclusions

Once the development of the project is finished, the moment arrives to make an assessment of what has been achieved and what has been learned.

The objective of this project was to implement different devices that obtain biomedical signals, which we can affirm has been achieved by measuring a total of four signals. In addition, it has been possible to unify the connection of all the sensors in a single board and a graphic interface has been developed that facilitates the visualization of the data.

As a result, knowledge on the subject of embedded systems has been expanded, as well as their capabilities and possible uses in other areas.

Although it is a functional system, it should not be used for medical diagnostic purposes since it does not use professional material, but rather sensors that are mostly intended for another purpose and have been adapted. This means that reliability is not guaranteed in the results. For this reason and for it to be a system, as well as functional, also reliable, professional sensors developed for this purpose and with guaranteed quality should be used.

In summary and despite the difficulties, it has been possible to carry out the proposed project, which has allowed us to expand and capture the knowledge acquired in this stage, making us feel very satisfied with it.

Bibliografía

[1] Sparkfun, AD8232 Heart Rate Monitor Hookup Guide.

[2] He Instruments | TechPatient Cardio.

[3] Genomasur.com, Capítulo 14: Sistema Respiratorio.

[4] Raspberry Pi 3 Model B+.

[5] Introduction to Arduino Nano, www.theengineeringprojects.com.

[6] Trastornos de la temperatura corporal, 2007.

[7] La capnografía en los servicios de emergencia médica — www.elsevier.es, 2009.

[8] Capnografía, la evolución en la monitorización del paciente crítico, 2013.

[9] GSR con Adruino, 2017.

[10] Capnografía — Wikipedia, The Free Encyclopedia, 2019.

[11] Electrocardiograma — Wikipedia, The Free Encyclopedia, 2019.

[12] Luis del Valle. `ds18b20 sensor de temperatura para líquidos con arduino`. *programarfacil.com*, 2018.

[13] Gregorio Patón Morales y Jorge Sánchez Carro Luis Barrado Muñoz, Santiago Barroso Matilla. `capnografía, la evolución en la monitorización del paciente crítico`. *zonates.com*, 2013.

[14] Inc Maxim Integrated Products. *`DS18B20 Datasheet Sonda de temperatura`*, 2018.

[15] Dr. F. Javier Montero Pérez. *APRENDER A INTERPRETAR EL ELECTROCARDIOGRAMA*. ELSEVIER, 2015.

[16] SprintIR. *SprintIR^{TM} Datasheet High Speed Carbon Dioxide Sensor*.

[17] SprintIR. *GSS Sensor User's Manua*.

[18] Cristian Veloso. www.electrontools.com, ARDUINO UNO PINOUT DIAGRAMA. 2016.

Apéndice A

Código fuente del programa para Arduino

```
1
2 #include <OneWire.h>
3 #include <DallasTemperature.h>
4 #include <SoftwareSerial.h>
5 // software serie : RX = pin digital 10, TX = pin digital 11
6 SoftwareSerial portOne(10, 11);
7 char buffer[50];
8 int co2 = 0;
9
10 //Temperatura
11 float temp = 0;
12 OneWire ourWire(2);//Se establece el pin 2 como bus OneWire
13 DallasTemperature sensors(&ourWire);
14
15 //ECG
16 int _ecg = 350; //valor base
17
18 //EDA
19 int eda = 0;
20
21 void setup() {
22   delay(1000);
23   Serial.begin(9600);
24   //CO2
25   portOne.begin(9600);// Start software serial port
26   //Temperatura
27   sensors.begin();   //Se inicia el sensor
28 }
29
30 void loop() {
31   //Temperatura
32   sensors.requestTemperatures();
33   temp = sensors.getTempCByIndex(0);
```

```
34    //EDA
35    eda = analogRead(A3);
36    //CO2
37    portOne.listen();
38    //Lectura de respuesta serial
39    // empty buffer
40    buffer[0] = '\0';
41
42    /**
43    * El sensor de CO2 tiene mayor retardo por su comunicacion paralela
44    * y al no tener opcion de multiproceso, ocasiona que se retrase la
       obtencion
45    * de datos del resto de sensores, unicamente relevante para el ECG, por lo
       que
46    * se intercala su ejecucion en la espera del sensor de CO2, obteniendo un
       resultado
47    * de mayor continuidad del ECG.
48    **/
49    for(int i = 0; i<=50;i++){
50      _ecg = analogRead(A0);
51      showData();
52      delay(1);
53    }
54    int idx = 0;
55    while(portOne.available()){
56      buffer[idx++] = portOne.read();
57    }
58    buffer[idx] = '\0';
59    co2 = atoi(&buffer[3]);
60
61    //pide medida
62    portOne.print("Z");
63    portOne.print("\r\n");
64
65    _ecg = analogRead(A0);
66    showData();
67  }
68
69  void showData(){
70    Serial.print(_ecg);
71    Serial.print(",");
72    Serial.print(temp);
73    Serial.print(",");
74    Serial.print(co2);
75    Serial.print(",");
76    Serial.println(eda);
77  }
```

Apéndice B

Código fuente de la interfaz gráfica

```
import processing.serial.*;

Serial myPort;// The serial port

//pantalla
static final int screenH = 600;
static final int screenW = 800;

 // cuadrante, posX, posY, valor min y max de la variable
Cuadrante c1 = new Cuadrante(1,screenW/2, screenH/2, 100, 700);
Cuadrante c2 = new Cuadrante(2,0, screenH/2, 20, 60);
Cuadrante c3 = new Cuadrante(3,0, 0, 0, 1000);
Cuadrante c4 = new Cuadrante(4,screenW/2, 0, 0, 200);
PFont font;

//calculo BPM
float currentBPM=0;
int beat_old = 0;
float threshold = 440.0;  //Threshold at which BPM calculation occurs
boolean belowThreshold = true;

void setup () {
  // set the window size:
  size(800, 600);

  // List all the available serial ports
  println(Serial.list());
  // Open whatever port is the one you're using.
  myPort = new Serial(this, Serial.list()[0], 9600);
  // don't generate a serialEvent() unless you get a newline character:
  myPort.bufferUntil('\n');
  //color fondo negro
  background(0);
  //color lineas blancas
  stroke(0xff);
  font = createFont("Arial", 12, true);
```

```
37   //font = createFont("Serif.bold", 12, true);
38 }
39
40
41 void draw () {
42   //division pantalla
43   line(0, screenH/2, screenW, screenH/2);
44   line(screenW/2, 0, screenW/2, screenH);
45
46   if ( myPort.available() > 0) {
47     String inString = myPort.readStringUntil('\n');
48
49     if (inString != null)
50     {
51       // trim off any whitespace:
52       inString = trim(inString);
53       String valores[] = split(inString,',');
54       try{
55         //ECG
56         calculateBPM(float(valores[0]));
57         c1.update(float(valores[0]));
58         c2.update(float(valores[1]));//Temperatura
59         c3.update(float(valores[2]));//CO2
60         c4.update(float(valores[3]));//EDA
61       }catch(Exception e){
62         println("Error de lectura");
63       }
64     }
65   }
66   c1.drawLine();
67   c2.drawLine();
68   c3.drawLine();
69   c4.drawLine();
70
71   c1.drawScale();
72   c2.drawScale();
73   c3.drawScale();
74   c4.drawScale();
75
76   c1.drawData();
77   c2.drawData();
78   c3.drawData();
79   c4.drawData();
80
81   delay(1);
82 }
83
84
85 void calculateBPM (float value){
86   if (value > threshold && belowThreshold == true)
87   {
```

```
88      int beat_new = millis();    // get the current millisecond
89      int diff = beat_new - beat_old;    // find the time between the last two
        beats
90      currentBPM = 60000 / diff;    // convert to beats per minute
91      beat_old = beat_new;
92      belowThreshold = false;
93    }
94    else if(value < threshold)
95    {
96      belowThreshold = true;
97    }
98  }
99
100
101
102 public class Cuadrante {
103   int cuadrante=0; //numero de cuadrante
104   int posX=0; //pos X del cuadrante
105   int posY=0; //pos Y del cuadrante
106   int time=0; //linea de tiempo en pos X
107   float value=0; //Valor senal escalado (pos Y)
108   float old_value=0;//valor anterior senal escalado (pos Y)
109   float data; //Valor original de la senal
110   int min, max; //valores maximos y minimos de la senal
111
112   Cuadrante( int _cuadrante, int _posX, int _posY, int _min, int _max){
113     cuadrante = _cuadrante;
114     value = 0;
115     old_value = 0;
116     posX = _posX;
117     posY = _posY;
118     time = posX;
119     min =_min;
120     max = _max;
121   }
122   void update(float _data){
123    if(_data<=max && _data>=min
124    ){
125      old_value = value;
126      try{
127        value = map(_data, min, max, 0, 300);//Escala valor
128      }catch(Exception e){
129        println("Error de lectura");
130      }
131      data = _data;
132    }
133   }
134   void drawLine(){
135     switch(cuadrante){
136       case 1:
137         stroke(240, 50, 0);
```

```
138         break;
139       case 2:
140         stroke(0, 255, 255);
141         break;
142       case 3:
143         stroke(0, 255, 0);
144         break;
145       case 4:
146         stroke(255, 255, 0);
147         break;
148     }
149     line(time-1, screenH - posY - old_value, time, screenH - posY - value);
150     stroke(0xff);
151     if (time >= posX + screenW/2) {
152       time = posX;
153       background(0);
154     }
155     else {
156       // increment the horizontal position:
157       time++;
158     }
159   }
160
161   void drawData(){
162
163     fill(0);
164       rect(posX+screenW/2-90, ((posY-screenH/2)*(-1)), 90, 50);
165       fill(0xff);
166       textSize(12);
167       switch(cuadrante){
168         case 1:
169           text("BPM:\n"+currentBPM, posX+screenW/2-90+5, ((posY-screenH/2)
    *(-1))+20);
170           break;
171         case 2:
172           text(data+"C", posX+screenW/2-90+5, ((posY-screenH/2)*(-1))+20);
173           break;
174         case 3:
175           text(data+"ppm", posX+screenW/2-90+5, ((posY-screenH/2)*(-1))+20);
176           break;
177         case 4:
178            String state="";
179           if(data<10){
180             state="Disconnected";
181           }else if(data>10 && data<75){
182             state="OK";
183           }else{
184             state="Warning!";
185           }
186           text(state+"\n"+data, posX+screenW/2-90+5, ((posY-screenH/2)*(-1))
    +20);
```

```
187             break;
188         }
189     }
190     void drawScale(){
191       int div = 8; //numero de divisiones en la escala
192       float valor = min;
193       for(int i=screenH-posY;i>screenH/2-posY;i-=screenH/2/div){
194         line(posX+18, i, posX+22, i);
195         textFont(font);
196         textSize(9);
197         text(parseInt(valor), posX, i+3);
198         valor += (max-min)/div;
199       }
200     }
201 }
```